IV Workshop sobre Software Livre – WSL 2003

Porto Alegre, RS - Brasil
5, 6 e 7 de junho de 2003

Anais

Editores
Denise Bandeira da Silva
Lisandro Zambenedetti Granville

Organização
FEEVALE
PUC-RS
UFRGS
UFSM
ULBRA
UNISINOS

Promoção
Sociedade Brasileira de Computação - SBC

Capa
Veraz Comunicações

Impressão
Editora Evangraf Ltda

Tiragem
4.000 exemplares

IV Workshop sobre Software Livre – WSL2003
(2. : 2003 05 a 06 de junho: Porto Alegre, RS)

Anais / IV Workshop sobre Software Livre – WSL2003; editores
Denise Bandeira da Silva e Lisandro Zambenedetti Granville, Porto Alegre, 05 a 06 de junho de 2003. Porto Alegre : Sociedade Brasileira de Computação, 2003.
162p. ; 23cm

1. Software 2. Software Livre 3. Programa I. Título II. Silva, Denise Bandeira. III. Granville, Lisandro Zambenedetti.

Endereço:
Editora Evangraf Ltda
Rua Waldomiro Schapke, 77 - Porto Alegre, RS - Brasil
Telefone (51) 3336-0422

Prefácio

O **Workshop sobre Software Livre** (WSL) é um evento que ocorre junto ao **Fórum Internacional Software Livre** (FISL). O FISL, que ocorreu pela primeira vez em maio de 2000, tem como principal objetivo discutir e divulgar o uso de software livre nos mais diversos segmentos da sociedade. Dentro desse contexto, o WSL tem papel fundamental por apresentar um panorama do desenvolvimento de software livre através de trabalhos e pesquisas que têm sido desenvolvidos pela comunidade acadêmica e científica. O WSL constitui-se, portanto, em uma oportunidade para que professores, alunos e pesquisadores apresentem trabalhos que estão sendo desenvolvidos em seus centros de pesquisa e universidades.

Neste ano, em sua 4ª. edição, o WSL obteve um recorde na submissão de trabalhos. Foram enviados para avaliação 67 artigos, dos quais 33 foram selecionados. Os processos de submissão, avaliação e seleção foram realizados de forma totalmente eletrônica através do sistema de gerência de eventos EDAS, que é um sistema desenvolvido com software livre. Cada artigo submetido foi avaliado pelo comitê de programa composto por professores e pesquisadores de diversas universidades. O trabalho do comitê de programa foi árduo devido ao número de artigos submetidos e, principalmente, à excelente qualidade dos trabalhos enviados. Nestes anais são apresentados os 33 trabalhos selecionados para o WSL 2003. Tais trabalhos também são apresentados nas sessões técnicas do WSL.

Gostaríamos de agradecer aos autores que submeteram seus trabalhos e colaboraram elevando a qualidade técnica do evento. Agradecemos também ao corpo de revisores do comitê de programa, sem os quais a seleção de artigos teria sido impossível de ser realizada. Cremos que os trabalhos aqui apresentados mostram um panorama fiel do desenvolvimento e pesquisa em software livre, e como tal, contribui para a disseminação dos conhecimento adquiridos com o software livre na sociedade.

Denise Bandeira da Silva, UNISINOS
Coordenadora do Comitê de Organização

Lisandro Zambenedetti Granville, UFRGS
Coordenador do Comitê de Programa

Porto Alegre, junho de 2003

Comitê de Organização

Alexandre Moretto Ribeiro, UCS
César Augusto Fonticielha de Rose, PUC-RS
Denise Bandeira da Silva, UNISINOS - Coordenadora
Luciana Porcher Nedel, UFRGS
Roland Teodorowitsch, ULBRA

Comitê de Programa

Alexandre Moretto Ribeiro, UCS
Cláudio Fernando Resin Geyer, UFRGS
Daniel Nehme Muller, ULBRA
Daniela Leal Musa, UFRGS
Denise Bandeira da Silva, UNISINOS
Edgar Meneghetti, UCS
Elgio Schlemer, ULBRA
Fabio Zschornack, UFRGS
Fernando Santos Osório, UNISINOS
Francisco Assis Moreira do Nascimento, ULBRA
Gerson Geraldo Homirch Cavalheiro, UNISINOS
João Batista Oliveira, PUC-RS
João Cesar Netto, UFRGS
José Palazzo Moreira de Oliveira, UFRGS
Leandro Krug Wives, FEEVALE
Lisandro Zambenedetti Granville, UFRGS - Coordenador
Luciano Paschoal Gaspary, UNISINOS
Luciana Porcher Nedel, UFRGS
Manuel Menezes de Oliveira Neto, UFRGS
Maria Janilce Bosquiroli Almeida, UFRGS
Marinho Pilla Barcelos, UNISINOS
Mariusa Warpechowski, UFRGS
Paulo Roberto Ferreira Jr., FEEVALE
Renata Zanella, UFRGS
Roland Teodorowitsch, ULBRA
Simone André da Costa, UNISINOS
Taisy Weber, UFRGS

Sociedade Brasileira de Computação

Diretoria

Presidente:	Flávio Rech Wagner, UFRGS
Vice-Presidente:	Luiz Fernando Gomes Soares, PUC-Rio
Administrativa e Finanças:	Taisy Silva Weber, UFRGS
Eventos e Comissões Especiais:	Ana Teresa de Castro Martins, UFC
Educação:	Marcos José Santana, USP/São Carlos
Publicações:	Claudia Maria Bauzer Medeiros, UNICAMP
Planejamento e Programas Especiais:	Robert Carlisle Burnett, PUC-PR
Secretarias Regionais:	Aleardo Manacero Jr., UNESP
Divulgação e Marketing:	Sérgio Cavalcante, UFPE
Regulamentação da Profissão:	Roberto da Silva Bigonha, UFMG
Eventos Especiais:	Ricardo de Oliveira Anido, UNICAMP

Conselho

Mandato 2001-2005
Ana Carolina Salgado, UFPE
Paulo Cesar Masiero, USP/São Carlos
Rosa Maria Vicari, UFRGS
Sergio de Mello Schneider, UFU
Tomasz Kowaltowski, UNICAMP

Mandato 1999-2003
Daltro José Nunes, UFRGS
Silvio Romero de Lemos Meira, UFPE
José Carlos Maldonado, USP/São Carlos
Therezinha Souza Costa, PUC-Rio)
André Carlos P. de Leon F. de Carvalho, USP/São Carlos

Suplentes - Mandato 2001-2003
Itana Maria de Souza Gimenes, UEM
Jaime Simão Sichman, USP
Miguel Jonathan, UFRJ
Raul Sidnei Wazlawick, UFSC

Secretaria Regional Rio Grande do Sul

Secretária: Luciana Porcher Nedel, UFRGS

Sumário

Palestras do Fórum Internacional Software Livre *08*

Um estudo para adoção de software livre na UCS *31*

Consolidação de Servidores e Serviços para a Comunidade Adminstrativa e Acadêmica da UNIJUI usando Linux em Mainframe IBM/s390 ..... *35*

Proposta de uma Política de Software Livre em Instituições de Ensino Superior: Um Estudo de Caso *39*

Software Livre suportando ambientes científicos de alta tecnologia *43*

Implementação de uma rede de computadores para ensino de física *47*

Avaliação como Forma de Incentivo ao Desenvolvimento de Software Livre em Ambiente Acadêmico *51*

ENSINET: Uma solução Software Livre para Apoio ao Ensino Semi-Presencial utilizando a Internet *55*

CRIANET: uma plataforma de software para crianças na Internet *59*

ETC - Editor de Texto Coletivo - Um software livre para auxiliar a escrita coletiva através da Web *63*

Software Livre nas pequenas e médias empresas - Tecnologia e Oportunidades *67*

Software Livre como Alternativa a Processos de Apuração de Dados em ONGs Prestadoras de Serviços *71*

GNU/Linux em máquinas IPF: Promessa ou realidade? *75*

Gerenciamento de Processos em Controladores Industriais usando XML *79*

Astrha/E - Ambiente Java/XML que Implementa em Hiper-Animações estruturadas por Máquinas de Mealy *83*

A Free Software for the Development of Adaptive Automata *87*

Aplicações SCADA utilizando GNU/Linux *91*

GoboHide: Uma Solução Flexível e Escalável para Inodes Ocultos no Kernel Linux *95*

Autenticação de Senhas Faladas Dependente de Usuário *99*

DiretoJ2EE: uma Modelagem J2EE para o Direto *103*

Módulo Agenda do DiretoGNU: Reestruturação das Classes e Adaptação das Interfaces *107*

Transformando a Informação Pública pelo Software Livre. Um olhar sociotécnico. *111*

Estudo Comparativo Entre Ferramentas Gratuitas Versus Proprietárias em um Processo de Desenvolvimento de Software Orientado a Objetos *115*

Um Modelo Espacial para Integração dos Dados da Saúde Implementado em Software Livre *119*

Estudo do Processo de Migração de Software Proprietário para Software Livre para Aplicações Web *123*

SAAP: Desenvolvendo e Expandindo Sistemas Usando Software Livre *127*

Implantação do Boot Remoto no Laboratório de Informática da Universidade de Caxias do Sul *131*

Open Source Software on Web-Based Education *135*

HackAcademia: Uma Experiência com Preparação de Novos Desenvolvedores para a Comunidade Software Livre *139*

Kiwi: Um Framework Alternativo para Aplicações com Interface Gráfica *143*

Desenvolvimento de um sistema de administração acadêmica para IES utilizando Software Livre *147*

Modelagem e Implementação do Banco de Dados do Projeto ESCUNA usando TCM e PostgreSQL *151*

Distribuição GoboLinux *155*

Portabilidade e flexibilidade em software livre: a experiência do GoboLinux *159*

A comunidade Rau-Tu e O sistema Nou-Rau: Armazenamento e indexação de documentos digitais

Humberto Sturiale Sartini
humberto@onda.com.br
Notícias Linux
Palestrante
Marcelo de Gomensoro Malheiros
mgm@valedofuturo.unicamp.br
Vale do Futuro
Palestrante
Eduardo Bacchi Kienetz
eduardo@noticiaslinux.com.br
Notícias Linux

O Sistema Rau-Tu possibilita que um time de colaboradores possa responder perguntas publicadas dinamicamente através do mesmo, cobrindo diversas áreas de conhecimento, denominadas tópicos. O usuário final (o qual publica suas perguntas/dúvidas) é notificado por e-mail da resposta de sua pergunta. Através do endereço provido no e-mail de notificação o usuário lê a resposta e faz sua avaliação de acordo com o grau de satisfação. Apenas respostas consideradas satisfatórias são armazenadas no banco de dados para posterior consulta, sendo que as insatisfatórias são automaticamente recolocadas à disposição para serem respondidas. Com isso somente respostas consideradas úteis são mantidas no banco de dados, o que agrega mais valor ao sistema.

O sistema Nou-Rau tem por objetivo implementar um sistema online para arquivamento e indexação de documentos, provendo acesso controlado e mecanismos eficientes para busca. São metas do sistema Nou-Rau: possibilitar o armazenamento de qualquer tipo de documento; manter informações básicas sobre cada documento, possibilitando a inclusão de dados mais específicos quando necessário; prover um mecanismo de busca que pesquise tanto nas informações quanto no conteúdo dos documentos; controlar o upload somente de documentos desejados e prover mecanismo para aprovação de documentos submetidos; e possibilitar intervenção externa para verificação de vírus. Somente documentos aprovados pelo responsável do tópico são publicados no sistema.

ABRASOL: Associação Brasileira das Empresas de Software Livre

Sandro Nunes Henrique
sandro@poa.conectiva.com.br
Conectiva

Administrando Escolas com Software Livre

Jairo França
Mauro Flores

A palestra apresenta o sistema Procergs Escola - Sistema de Gerenciamento Escolar, cujos benefícios são o gerenciamento racional do serviço de controle da secretaria; agilização, confiabilidade e qualificação dos serviços prestados tanto ao aluno quanto a comunidade escolar; aumento da produtividade da secretaria da escola; possibilidade de integração do estabelecimento com órgãos oficiais, como Secretaria da Educação e Ministério da Educação; rapidez na localização de dados e emissão automática de relatórios e documentos.

Alfabetização Infantil e Matematicamente Livre

Cristiane Edna Camboim
cris.camboim@bol.com.br
GNUrias
Ana Paula Fiegenbaum
ana@univates.br
UNIVATES
Josi Graciela Petter
josi@univates.br
GNUrias

"Pesquiso para conhecer o que ainda não conheço e comunicar ou anunciar a novidade" (Paulo Freire).

Aprender a aprender na era de tecnologias e informações é condição imprescindível para a formação de um bom profissional. Desta forma, o uso do computador como ferramenta de ensino-aprendizagem tem sido de suma importância para educadores em todas as áreas de conhecimento.

A formação do professor é um aspecto a ser pensado neste momento, pois as tecnologias interferem na função docente em termos de construção, produção e comunicação do conhecimento.

Acreditamos que o uso de software livre proporciona a democratização de saberes, possibilitando a criação de ambientes onde se possa refletir sobre a própria aprendizagem.

Teremos dois momentos distintos e interligados: a alfabetização infantil e o ensino-aprendizagem de matemática.

Na área de ensino de matemática serão relatadas experiências que estão sendo realizadas em âmbito regional e nacional e como se comportam estes professores frente ao uso de novas tecnologias.

Para a alfabetização infantil, tutoriais desenvolvidos por acadêmicas da Univates, já utilizados no Fórum Social Mundial.

Esta palestra abordará um dos aspectos fundamentais existentes na "Filosofia do Software Livre": compartilhar, considerando que a utilização de ambientes informatizados com fins educativos, exige mudanças nos modos de compreender o ensino e na prática pedagógica dos professores no cenário nacional.

Alta Disponibilidade - Failover e Load Balance

Sulamita Garcia
sulamita@4linux.com.br
LinuxChix Brasil
Mario Bonilla
miope@linux.org.uy
UYLUG - Grupo de Usuários Linux del Uruguay

A invasão tecnológica em todas as áreas da sociedade tem ajudado a melhorar e otimizar muitas tarefas do nosso dia a dia. Porém aumenta com isto a dependência a estes sistemas, que causam grandes transtornos em caso de paradas. Uma parada de alguns minutos pode causar desde filas e irritação à perda de somas significativas por falta de disponibilidade dos sistemas.A Alta Disponibilidade em sistemas de software livre é uma alternativa simples, eficiente e de baixo custo para garantir o atendimento ininterrupto aos usuários. Nesta palestra demonstraremos conceitos envolvidos em um ambiente de Alta disponibilidade, suas vantagens, desvantagens, aplicações e softwares disponíveis para tal. Aproveitaremos para falar um pouco sobre balanceamento de carga para melhor desempenho dos serviços e aproveitamento.

Auditoria de Segurança com Software Livre

Evandro Oliveira
evandro@pbh.gov.br
PRODABEL

Banco de Dados pela MySQL AB

David Axmark
david@mysql.com
MySQL AB

A palestra irá iniciar com a história do MySQL e então continuar com uma apresentação das funcionalidades atuais do MySQL (versão 4.0/4.1). MySQL tem nestes novos recursos objetivado entrar no mercado de Banco de Dados Empresariais. MySQL já esta sendo utilizado em aplicações de missão crítica em empresas como Yahoo!, Cisco e Google.

A MySQL AB utiliza um esquema de licenciamento duplo, onde o mesmo código fonte é disponibilizado sobre a licença GPL e sobre uma licença não GPL, comercial. Esta única forma de misturar um negócio de software comercial com o software livre será mencionada na palestra.

Com mais de 28000 downloads de servidor por dia somente através do site www.mysql.com e um número estimado de 4 milhões de instalações do Banco de Dados MySQL, este é um dos bancos de dados SQL mais utilizados no mundo.

Caos Criativo... Cultura, Liberdade e Software Livre

Sérgio Amadeu
Marcelo D'Elia Branco

Comparando PostgreSQL, MySQL e Interbase/Firebird

Fernanda Weiden
fernandaweiden@bol.com.br
LinuxChix Brasil

Traçar um paralelo entre os mais utilizados bancos de dados livres do mercado.Falar da historia, recursos, o porque cada um desses SGBD tem conquistado tanto a confiança de equipes de desenvolvimento. Quais os pontos fortes e fracos de cada uma das três alternativas, PostgreSQL, Firebird e MySQL.

Consórcio de Desarrolladores y Usuários de Software Libre - UNESCO Uruguai

Cláudio Menezes
cmenezes@unesco.org.uy
UNESCO Montevidéu

Criptografia Libre

Enrique A. Chaparro
echaparro@vialibre.org
Vía Libre

Desenvolvimento de Software Livre na Unicamp

Rubens Queiroz de Almeida
queiroz@unicamp.br
UNICAMP

A educação é o setor, segundo diversos especialistas, que oferece o maior potencial de crescimento nos próximos anos. A iniciativa privada vem explorando diversas possibilidades nesta área, estabelecendo parcerias e lançando raízes em um mercado em

expansão e com necessidades nas mais diversas áreas, desde a educação formal a cursos profissionalizantes.

O uso do computador na educação, seja em cursos totalmente à distância como em cursos onde o computador é apenas um item a mais, propiciando um melhor acesso a professores, material didático e outras atividades, é sem dúvida alguma imprescindível.

A tecnologia e a infra-estrutura necessária para o emprego de computadores na educação, dependendo da solução adotada, pode se tornar proibitiva, gerando mais um fator de exclusão. Somente instituições com elevados recursos financeiros para investimento nesta área poderiam oferecer a seus alunos acesso a esta tecnologia fundamental nos dias de hoje.

O software livre, embora não seja uma solução universal, pode contribuir significativamente para a disseminação e uso em larga escala de soluções eficientes e de baixo custo para a educação, à distância e mediada por computador.

A quantidade de relatos do uso bem sucedido do software livre em escolas e universidades são prova da viabilidade desta alternativa. Apesar do preconceito em geral contra o emprego de soluções baseadas em software livre para a educação em geral, os casos de sucesso são numerosos e representam uma prova eloqüente de sua viabilidade.

Desenvolvimento para Governos com Padrões Multiplataformas

Bruno de Souza
bruno.souza@javaman.com.br

DiretoGNU - Correio Eletrônico, Agenda e Catálogo Corporativo

Ricardo Balinski
balinski@via-rs.net
PROCERGS

O Direto é uma ferramenta online de correio, agenda e catálogo desenvolvida pela PROCERGS utilizando os princípios do software livre. Foi desenvolvido com o principal objetivo de fornecer uma solução de baixo custo para ser implantada em todo o Governo do Estado do Rio Grande do Sul.

A apresentação mostrará a evolução do produto através de seu histórico até o lançamento de sua nova versão, o Direto 3.0. Esta nova versão surgiu a partir da evolução das tecnologias da Web e da necessidade de suporte a novas funcionalidades. Possui uma interface totalmente remodelada, com as telas reestilizadas e as informações apresentadas de forma mais clara e consistentes.

Além da nova versão serão apresentadas outras informações importantes, como o uso do Direto no Governo do Estado do Rio Grande do Sul, a infra-estrutura utilizada e outras informações relacionadas.

Dot Learning - MIT

Roberto de Mello
rmello@cc.usu.org
MIT

Educação e Software Livre

Léa Fagundes
LeaFagun@vortex.ufrgs.br
UFRGS

Estratégia da HP para Software Livre

Bdale Garbee
bdale@gag.com
HP

O projeto Debian é um exemplo muito bem sucedido do modelo de desenvolvimento comunitário aplicado ao Software Livre. Nesta apresentação, Bdale irá se utilizar a sua perspectiva única e longa experiência com Debian para explicar como decisões chave na história do projeto diferenciaram Debian de outras distribuições e conduziram ao sucesso do projeto. Ele irá explicar como a comunidade Debian produz a distribuição Debian GNU/Linux e falar um pouco sobre mudanças prováveis no futuro. Para ilustrar estas idéias Bdale irá discutir o relacionamento da HP com Debian, e mostrar como a GP utiliza Debian GNU/Linux no desenvolvimento de sistemas Itanium. Assistir a esta palestra irá ajudar a indivíduos e empresas a entender como ser bem sucedidos trabalhando e contribuindo com o projeto Debian.

Extreme Programming - Supere o Medo

Humberto Soares
FALTA@EMAIL
Objetive Solutions
Klaus Wuestefeld
klaus@objective.com.br
Objetive Solutions

Extreme Programming (XP) é uma metodologia de desenvolvimento de software que prima por qualidade e agilidade.

A maioria das regras de XP causa polêmica à primeira vista e muitas não fazem sentido se aplicadas isoladamente. É a sinergia de seu conjunto que sustenta o sucesso de XP, encabeçando uma verdadeira revolução de metodologias ágeis. Por não ter um centro

autoritário, o espírito da comunidade ágil é muito similar ao espírito do mundo do software livre.

ALGUMAS CARACTERÍSTICAS DE XP

Mudanças de requisitos são bem-vindas, a qualquer momento durante o projeto; todo código de produção é programado em duplas;
A medição de progresso num projeto XP é feita através de funcionalidade em execução e não através de documentação obscura para o cliente;
Diferente do que ocorre no RAD, num projeto XP o código está sempre na sua forma mais manutenível, mais extensível e mais clara possível;
O cliente pode começar a testar o produto, tipicamente, já a partir da terceira semana de desenvolvimento;
Qualquer desenvolvedor pode trabalhar em qualquer código fonte (nenhum fonte tem dono único);
Nenhuma documentação de projeto é necessária durante o desenvolvimento;
Todo código é testado através de scripts de teste automatizado;
Todos os scripts de teste são desenvolvidos ANTES do respectivo código;
O design é feito DEPOIS da implementação e não antes;
XP requer uma disciplina férrea por parte da equipe. Mesmo assim, programadores gostam dela por não ser uma disciplina burocrática;
XP dá aos gerentes incomparável agilidade na alocação e desalocação de desenvolvedores em seus projetos.

Com esta breve exposição é possível ter uma noção de XP e, mais importante, perceber que existe alternativa ao marasmo metodológico elitista que tem dominado a indústria de software nos últimos tempos.

URL: http://www.xispe.com.br

Ferramentas Livres para Java

Fernando Lozano
fernando@lozano.eti.br

A sinergia entre o mundo Java e o software livre, com seus objetivos de abertura, padronização e independência de plataforma ou fornecedor gerou uma série de ferramentas incrivelmente úteis a projetos de todos os portes. Várias delas já são consideradas padrões de fato para a indústria e incorporadas a produtos consagrados como o Websphere, JBuilder e Oracle iAAS. Na verdade, o arsenal de software livre para o desenvolvedor Java é tão rico que torna viável desenvolver e implantar soluções baseadas unicamente em software livre. Esta palestra apresenta as principais ferramentas, como o compilador GCJ, os IDEs Eclipse e NetBeans, o gerenciador de configurações Ant, o gerador de código XDoclet, Os frameworks JUnit e Struts, e os servidores Tomcat, JBoss e HSQLDB.

Além é claro de mencionar a integração com outros softwares livres consagrados como o Linux, Apache, MySQL, PostgreSQL e CVS.

FreeBSD - IPFW 2

Patrick Tracanelli
eksffa@freebsdbrasil.com.br
FreeBSD Brasil
Jean Mello
FreeBSD Brasil

O IPFW2 - IPFIREWALL 2 - foi rescrito com o princípio básico de oferecer maior flexibilidade e performance à criação e processamento de suas regras. Como resultado, a performance chega a ser 250% maior em determinados ambientes, e a flexibilidade do novo código inspirou várias melhorias no IPFIREWALL. Todo o (há muito) já existe suporte a regras Stateful e Dinâmicas (criadas a partir de regras Stateful ou Stateles) foi aperfeiçoado, oferecendo melhoria notável às limitações por destino ou origem, flexíveis por mascaras de subrede. Uma tabela de origem reversa foi adicionada, facilitando controle da técnica que evita pacotes com endereço forjado (IP Spoof), adicionados conjuntos de regras e uma um grande número de novas funcionalidades do mais clássico subsistema de Firewalling do mundo Open Source.

FreeBSD 5

Diego Linke
gamk@gamk.com.br
FUG Brasil

O FreeBSD 5.0 marca o lançamento da primeira versão de uma nova série do FreeBSD, em desenvolvimento há mais de 2 anos. É a maior quantidade de novos códigos e novas tecnologias lançadas ao mesmo tempo em um sistema Open Source. Conta com novas características estruturais reimplementadas a partir de seu princípio, notadamente o suporte à multiprocessamento simétrico, agendamento de threads por kernel e implementação das múltiplas APIs de segurança especificadas no POSIX.1e, projeto conhecido como TrustedBSD, e uma série de novas características nos subsistemas de armazenamento (GEOM, UFS2, Snapshots, Background Fsck) e aperfeiçoamento do SoftUpdates, bem como diversas novas implementações ao sistema de redes (como Fast_IPSec, IPFW2, suporte a criptografia por hardware), e uma série de outras novidades.

FreeNMS - Sistema de Gerenciamento de Rede em Software Livre - PUC-RS

Palestrante
Jorge Guedes Silveira
guedes@ee.pucrs.br
PUC-Rio
Palestrante
Marcelo Azambuja
azambuja@ee.pucrs.br
PUCRS

Nesta palestra será apresentado o sistema FreeNMS - Free Network Management System (www.freenms.org), um Sistema Software Livre Gerenciador de Redes e Nível de Serviço (SLM) 100% Web-based.

Este projeto é desenvolvido pelo Laboratório ReMAV MetroPoa da PUCRS e possui o apoio de duas grandes operadoras Telecom.

Neste sistema, são utilizados alguns dos principais softwares condizentes com as diretrizes do Software Livre, tais como o Sistema Operacional GNU-Linux, servidor Web Apache, Bancos de Dados MySQL e PostGreSQL, as linguagens C++, Perl e PHP.

Como resultados temos a implementação de uma ferramenta que visa ajudar nas tarefas de gerenciamento de nível de serviço e a convicção da necessidade e o espaço atualmente existente para trabalhos teóricos e prático nesta área.

Gnome pelo Seu Criador

Miguel de Icaza
miguel@ximian.com
GNOME Foundation

Grupo de Empresas de Software Livre do Seprorgs

Gisele Machado de Oliveira
gisele@seprors.com.br
FENAINFO

Imprimindo com CUPS e Foomatic - Com Demonstrações ao Vivo

Till Kamppeter
till.kamppeter@gmx.net
LinuxPrinting

Imprimir é uma das tarefas diárias mais complicadas na informática -- em todos os sistemas operacionais.

Por muito tempo a impressão em Unix e GNU/Linux era baseada em tecnologia dos anos 70, usando o antigo "Line Printer Daemon" (LPD). LPD foi inicialmente desenvolvido para impressoras que só sabiam imprimir texto ASCII. Hoje em dia as impressoras são bem diferentes. Eles não só imprimem texto: Cores, gráfica, fotos, tipos diferentes de papel, grampear, encadernação, e mais. Mas ainda tem implementações deste sistema não suportando estas opções e difícil a administrar em redes instaladas em muitos lugares.

Dois projetos dos últimos três anos facilitaram a vida dos usuários e dos administradores muito:
- O sistema de impressão (spooler) CUPS (Common UNIX Printing System) baseado no IPP (Internet Printing Protocol) e em PPDs (Postscript Printer Description)
- Foomatic do linuxprinting.org, uma infra-estrutura de integração de drivers com todos os sistemas de impressão

Estes projetos são muito úteis para todos os usuários, independente que t&em só um computador com uma impressora em casa ou uma empresa com milhares de computadores e impressoras.

CUPS fornece uma arquitetura para imprimir em redes que é muito fácil a usar: Se configurar uma impressora em um servidor, o servidor divulga a informação automaticamente e então os clientes CUPS podem usar esta impressora sem configurar nada neles. Com o IPP sendo baseado no protocolo HTTP o CUPS suporta padrões comuns da Internet, como SSL, LDAP, autenticação, e mais. Usando o conceito de arquivos PPD que foi inventado para descrever as capacidades de impressoras PostScript para os drivers Windows e Mac OS, o usuário tem acesso a toda a funcionalidade de todas as impressoras PostScript com o CUPS facilmente.

A parte principal de Foomatic é uma base de dados XML de mais que 1000 impressoras e mais que 200 drivers com informação que impressora funciona com que driver e como o GhostScript e outros filtros tem que ser chamados para tornar o PostScript produzido pelos aplicativos para o formato que a impressora entende. Destes dados são gerados PPDs que, junto com um filtro universal escrito em Perl, dão acesso a toda funcionalidade dos drivers com todos os sistemas de impressão (CUPS, LPD, LPRng, GNUlpr, PPR, PDQ, CPS. e até impressão sem spooler).

A maioria das distribuições atuais de GNU/Linux usa CUPS e Foomatic, como Mandrake, Red Hat, Conectiva, SuSE, Debian, e outros, mas muitos usuários não sabem sobre isso e não utilizam toda a funcionalidade deste ambiente poderoso. Neste tutorial vai ser mostrado ao vivo como lidar com impressão nestes sistemas. O tutorial compõe-se de
- Configuração de filas de impressão locais
- Partilhar impressoras em redes
- Imprimir e tratar processos de impressão
- Resolver problemas

A programação exata depende da audiência, não é um problema tratar um tema mais intensivo, responder perguntas, e até conectar um notebook de um visitante.

Este tutorial é para todo usuário de GNU/Linux ou Unix que quer ter uma boa impressão de software livre também na impressão – usuários com uma impressora em casa e também administradores de redes.

Inclusão Digital com Software Livre: Experiências da Sociedade Civil - GNUTeen e Gemas da Terra

Joel dos Santos Raymundo
PROCEMPA
Claudimir Zavalik
FALTA@EMAIL
APOENA - Software Livre
Marco Aurélio de Figueiredo
marco@paradigm-computing.com
Paradigm

O Grupo GNUTeen é um grupo de usuários GNU/Linux que surgiu a partir da iniciativa de promover a inclusão digital de adolescentes que não possuíam acesso à Tecnologia da Informação. É um trabalho voluntário que conta com a ajuda de diversos colaboradores, recebe apoio institucional da Free Software Foundation e do PSL e é mantido através de doações da comunidade, empresas e instituições. Entre os planos do grupo, está a criação de novos grupos em outras localidades, formando uma "Rede GNUTeen". O Grupo irá apresentar sua experiência, relatando sua história e detalhando seus objetivos e realizações.

Inclusão Digital e Software Livre: Uma Visão Governamental

Rogério Santanna dos Santos
rogerio.santanna@planejamento.gov.br
SLTI - MP
Joel dos Santos Raymundo
PROCEMPA
Wagner Meira Jr.
meira@dcc.ufmg.br
UFMG

Internet no Brasil: Passado, Presente e Futuro

Arthur Pereira Nunes
Comitê Gestor
Ivan Moura Campos

J Banana - Framework MVC Open Source para Java/J2EE

Sandro Bihaiko
misato@gci.com.br
GCI Soluções

Produtividade com framework brasileiro open-source e livre. Apresentação da estrutura e componentes para acelerar construção de aplicações.

La Revolución Digital en España: Los Primeras Herramientas; Conectividad, Alfabetización Tecnológica y Software Libre

Juantomás García Molina
juantomas@lared.es
HispaLiNUX

LESP: Configurando Servidores de Forma Fácil

Anahuac de Paula Gil
anahuac@linuxview.com.br
LUG/PE

O LESP-CEL é uma ferramenta administrativa que permite o gerenciamento remoto de computadores com Gnu/Linux, através de celulares ou quaisquer tipo de dispositivo capaz de enviar e receber e-mails. Basicamente o que o LESP-CEL faz é filtrar mensagens eletrônicas e executar uma instrução, de acordo com os parâmetros indicados.

O objetivo é permitir total manuseio do Gnu/Linux por equipamentos portáteis que não possuem sistema operacional próprio. Esse é o caso dos celulares. Nem sempre se tem a disposição um computador ou um notebook para realizar tarefas remotamente.

Com o LESP-CEL é possível utilizar o serviço de e-mail de um celular com suporte a WAP para realizar qualquer atividade em um computador que tenha o GNU/Linux instalado.

O LESP-CEL esta inteiramente desenvolvido 1 shell script, utilizando alguns utilitários do sistema, como "cut", "sed", "tail", "head", "cat", "grep" e outros, que normalmente estão instalados em qualquer distribuição GNU/Linux. O LESP-CEL também possui uma série de controles de segurança para impedir o uso indevido dos seus recursos.

LIDS - Segurança com Software Livre

Leandro Márcio Bertholdo
berthold@penta.ufrgs.br
UFRGS

Durante a palestra serão abordados os seguintes temas:
* Pacotes e configurações possíveis para melhorar a segurança no S.O. Linux (LIDS Project).

* Características de segurança nativas ao FreeBSD (securelevel, controle de acesso ao sistemas de arquivos)
* O Conceito de Trusted Systems (Trustix e TrustedBSD)
* Novas direções e pesquisas em Segurança de Redes.

Lightning Talks: Tecnologias Zope e Python de forma descontraída e produtiva

Fabiano Weimar dos Santos
fabiano@x3ng.com.br
X3ng

Essa é uma iniciativa clássica das Python Conferences. Várias tecnologias do mundo Python e Zope, todas muito interessantes e apresentadas de uma forma nova, descontraída e produtiva.

Linux in Brazil

Augusto Campos
brain@matrix.com.br
Linux in Brazil

Linux Terminal Server Project (LTSP)

James McQuillan
jam@McQuil.com
LTSP

LinuxPrinting.org e Foomatic; A nova geração de imprimir com software livre

Till Kamppeter
till.kamppeter@gmx.net
LinuxPrinting

Fazer uma impressora moderna funcionar corretamente em sistemas operacionais livres é uma tarefa complicada, especialmente se precisa de acesso a toda funcionalidade da impressora.

Para facilitar isso o Grant Taylor, autor do "Printing HOWTO", criou uma base de dados de impressoras e de drivers. Esta base de dados, chamada "Foomatic" é situada em http://www.linuxprinting.org/, e cuja manutenção eu estou fazendo desde meados de 2001, contem entradas para todos os drivers livres conhecidos com as suas linhas de comando e todas as suas opções. Também toda impressora sobre que temos informações é listada com informações sobre como usá-la com software livre, que driver é melhor e a qualidade de suporte por software livre.

A base de dados é implementada em XML e tem scripts em Perl para automaticamente gerar arquivos PPD (Postscript Printer Description) e até filas de impressão para todos os sistemas de impressão livres: CUPS, LPD, LPRng, GNUlpr, PPR, PDQ, CPS, e impressão sem spooler. Com estas filas o usuário vai ter acesso a toda a funcionalidade de impressoras e vai ter o mesmo interface para todos os sistemas de impressão.

Este sistema agora é um padrão inoficial: Ele é utilizado nas distribuições Mandrake, Red Hat, Conectiva, SuSE Debian, e provavelmente outras e o linuxprinting.org é consultado por em torno de 10000 pessoas todo dia.

Nesta palestra vou falar sobre:
- A história do Foomatic - A estrutura da base de dados
- Como arquivos PPD são gerados com Foomatic?
- Como a impressão mesma funciona?
- Como o desenvolvimento de Foomatic vai continuar?

Esta palestra é para usuários e administradores de sistemas interessados em como o Foomatic funciona e também para quem quer participar no desenvolvimento do Foomatic.

Mulheres no Mercado de Trabalho de Software Livre

Ana Paula de Araujo
preta@univates.br
GNUrias
Fernanda Weiden
fernandaweiden@bol.com.br
LinuxChix Brasil
Loimar Vianna
loimar-vianna@procergs.rs.gov.br
PROCERGS
Paloma Costa
paloma_costa@yahoo.com.br
UNIVATES
Sulamita Garcia
sulamita@4linux.com.br
LinuxChix Brasil

Embora existam mais mulheres do que se pensa na área de tecnologia em geral, e do software livre, elas aparecem muito pouco. Esse debate pretende levantar de forma franca, questões sociais, de formação e psicológicas que influenciam em muito no fato de que as mulheres ainda não ingressam na área no mesmo numero que homens levantando sugestões.

Negócios Livres - Free Software Consortium

Jaco Aizenman
jaizenman@fsc.cc
FSC

O Software Livre na Mídia

Paulino Michelazzo
paulino@michelazzo.com.br
Quilombo Digital
Ricardo Andere de Mello
gandhi@quilombodigital.org
Quilombo Digital

"Quem não se comunica, se estrumbica" dizia o velho guerreiro Abelardo Barbosa, o Chacrinha. Ícone das tardes de sábado com sua liderança de audiência e seu jeito despojado de falar bem e mal de tudo e de todos, marcou a vida de muita gente e, principalmente, mostrou que a comunicação é a alma de todo o negócio.

O software livre há muito deixou de ser notícia de rodapé em veículos especializados para se tornar capa de revista que muitas vezes nada tem a ver com computação. Mas assim mesmo, existem grandes "gaps" de comunicação, aproveitados de forma primordial pelas outras vertentes existentes que, se apoiando nisto, mostram inverdades e informações deturpadas de sua real atividade e foco.

Pontos como campanhas de mídias, veículos de comunicação e ações de marketing são apresentados e debatidos, bem como mudanças que podem ser levadas a cabo em pouco espaço de tempo, com a finalidade de colocar o software livre em seu devido lugar de plataforma socialmente correta e financeiramente atrativa para qualquer um.

OpenOffice.org - Suite Office para Desktop

Claudio Ferreira Filho
filho@novanis.com.br
OpenOffice.org

O OpenOffice.org é um projeto de código aberto com a missão de criar, como uma comunidade, a suite Office internacional que rodará na maioria das plataformas e proverá acesso a todas as funcionalidades através de APIs de componentes de código aberto e com arquivos no formato XML.

Neste encontro, será apresentado o que é o OpenOffice.org, seus componentes, as inovações esperadas para a nova versão(1.1), bem como o trabalho da extensão brasileira

do projeto, o OpenOffice.org Projeto Brasil, seu trabalho e estrutura, além de uma abordagem sobre migração e números da suite no país.

Pão e Liberdade: Software Livre no Combate à Fome

Frei Betto
Presidência da República
Joel dos Santos Raymundo
PROCEMPA
Mario Luis Teza
mlteza@via-rs.net
DATAPREV

O Movimento Software Livre tem sido protagonista de muitas obras grandiosas pelo mundo afora. A NASA o utiliza para pesquisa e vôos espaciais, a Petrobrás, na busca de petróleo, bancos, como o Banrisul, para terem mais segurança e velocidade, o Departamento de Defesa dos EUA, para aumentar seu aparato de segurança. A Comunidade de Software Livre também tem auxiliado na busca de água em regiões inóspitas da África, em hospitais em Cuba, no Instituto do Câncer em São Paulo. Mas nenhuma dessas ações compara-se ao desafio do Governo do Presidente Lula.

A Comunidade de Software Livre brasileira está à altura do desafio que o Programa Fome Zero propõe. Tem experiência, amplitude, articulação internacional, base instalada nas universidades, empresas públicas, privadas, ONGs. Viveu a experiência do Governo do Rio Grande do Sul e de outros Estados, além da de dezenas de prefeituras.

Podemos saciar a fome de quem não tem o que comer com a mais alta liberdade tecnológica que a humanidade já desenvolveu.

PDC com Samba utilizando LDAP

Palestrante
Clodonil Trigo
clodonil@4linux.com.br
4 Linux

1)O que é PDC
2)Vantagens de um PDC
3)Origem do Samba
4)Implementando o PDC com Samba
5)Implementando o PDC com Samba via LDAP

Por que há tão poucas mulheres ativas na computação e em especial no GNU/Linux?

Fernanda Weiden
fernandaweiden@bol.com.br
LinuxChix Brasil
Sulamita Garcia
sulamita@4linux.com.br
LinuxChix Brasil

Não é raro nós, mulheres que trabalham na comunidade, ouvirmos esta interrogação. E em qualquer encontro, lista de discussão, nota-se a diferença entre números de homens e mulheres. Quais os motivos desta diferença? Mulheres simplesmente não se interessam? Se estas questões já passaram por sua cabeça, ou se despertaram curiosidade agora, esta palestra tem a proposta de explicar alguns motivos pelos quais ainda somos minoria. Questões sociais, de formação, e algumas especificas à comunidade de software livre. Esperamos esclarecer um pouco a questão, e lançar algumas sugestões sobre como modificar esta tendência.

Programação Shell Linux

Palestrante
Julio Neves
DATAPREV

Projeto Debian

Wichert Akkerman
wichert@cistron.nl; wichert@wiggy.net
Cistron

Exatamente como nos tempos antigos, o mundo dos dias modernos é repleto de perigos e de mal. Para sobreviver na Internet selvagem um indivíduo precisa estar alerta aos perigos que estão lá fora e como se proteger dos mesmos. Esta apresentação irá focar em como proteger um sistema Debian GNU/Linux observando as ferramentas e políticas presentes, bem como possíveis extensões

Projeto Hipatia

Diego Saravia
dsa@unsa.edu.ar
UTUTO
Enrique A. Chaparro
echaparro@vialibre.org
Vía Libre

Projeto Mozilla: Desenvolvimento Colaborativo e Ferramentas de Software

Christian Reis
kiko@async.com.br
Async

O Projeto Mozilla é um dos projetos de software livre mais complexos e interessantes na atualidade. Inicialmente baseado no código do que viria a ser o Netscape 5, e contando com um investimento grande da AOL para seu desenvolvimento, o projeto, sua política de desenvolvimento e as ferramentas que utiliza vêm evoluindo ao longo do tempo em direção a um processo organizado, aberto e muito interessante.

Esta apresentação descreve o processo de desenvolvimento, e oferece uma visão geral das ferramentas de software utilizadas, o que inclui Bugzilla, Tinderbox, Bonsai e LXR. É uma boa introdução para quem quer conhecer melhor o projeto ou descobrir como começar a contribuir.

Projetos de Lei Sobre Software Livre no Brasil

Sérgio Miranda de Matos Brito
sergiomirandabh@aol.com
Câmara Federal

Seminário Internacional de Governos e Software Livre

Sérgio Amadeu
Sergio.amadeu@planalto.gov.br
ITI
Rogério Santanna dos Santos
rogerio.santanna@planejamento.gov.br
SLTI - MP
Francisco Huertas Méndez
jesus.rubio@juntaextremadura.net
LinEx

Software Livre e OO

Alessandro Binhara
binharalista@yahoo.com.br
Rede Sol

Abordagem de conceitos básicos de orientação a objetos e uma visão geral do software livre aplicado a plataforma Java.

Software Livre no Ambiente Corporativo

Nelson Corrêa de Toledo Ferraz
nferraz@phperl.com
Quilombo Digital
Paulino Michelazzo
pmichelazzo@quilombodigital.org
Quilombo Digital

A cada ano que passa, o software livre ganha mais força, movido pela paixão de milhares de desenvolvedores e milhões de usuários que acreditam em uma ética segundo a qual o conhecimento não deve permanecer oculto, mas ser compartilhado.

Enquanto isto, grande parte da discussão sobre as vantagens ou desvantagens do software livre para o ambiente corporativo tem sido ao redor do custo, argumentação estimulada pelos fabricantes de softwares proprietários -- a quem interessa que os custos sejam analisados isoladamente, fora de um contexto estratégico.

Esta apresentação tem como objetivo mostrar, através de estudos de casos, que o software livre é uma fonte de vantagens estratégicas para as empresas, como a redução das barreiras de mudança, independência de fornecedor, segurança e confiabilidade na execução de processos e a possibilidade de diferenciação.

Soluções Corporativas na Web com Alternativas Livres

Giovani Spagnolo
giovani@webyes.com.br
WebYES!
Piero Josen
piero@webyes.com.br
WebYES!

Público Alvo: empresários em geral e gerentes de tecnologia.

Objetivos:
* Desmistificar os medos empresariais sobre software livre;
* Mostrar que empresas sérias estão desenvolvendo software livre;
* Apresentar as grandes empresas que estão migrando para software livre;
* Responder porque médias e pequenas empresas também podem usar softwares livres;
* Apresentar vantagens de sistemas web;
* Informar como migrar sistemas “off-line” para web;
* Informar como avaliar os custos de um software livre;
* Apresentar case do YES! Eventos, software livre usado no controle do IV FISL;
* Apresentar case do YES! Notícias, software livre usado no controle de imprensa do IV FISL.

Com a expansão da filosofia de software livre pelo mundo inteiro, as empresas também começam a avaliar como esta nova forma de fazer negócios pode impactar sobre suas atividades. Ainda assim, estas empresas enfrentam temores que as impedem de prosseguir em muitos de seus projetos com tecnologias livres. Aos poucos, muitos destes temores vão sendo desmistificados, enquanto outros vão sendo absorvidos pela comunidade de software livre, na tentativa de qualificar ainda mais os novos sistemas.

Analisando o avanço das tecnologias de Internet nos últimos anos, as empresas começaram a notar a necessidade de interagir com mais velocidade, na tentativa de competir com potenciais concorrentes. A migração de sistemas "off-line" para sistemas construídos em plataforma web permite que as empresas se beneficiem dessas inovações trazidas pela Internet, afim de estabelecer um novo padrão de comunicação. Os custos destas migrações deverão ser absorvidos pelas empresas de uma forma ou de outra, então, porque a escolha por software livre? A redução de custos com softwares livres dá-se principalmente através da inexistência de licenciamento do software. Uma vez livre, é possível utilizar cópias em quantas máquinas cliente ou servidor forem necessárias, sem custos extras, e por período indeterminado.

Para a organização e controle das informações necessárias à realização do IV Fórum Internacional do Software Livre, o Projeto Software Livre RS utilizou dois softwares livres produzidos pela WebYES! Internet Systems. O YES! Eventos e o YES! Notícias são softwares que, integrados, realizam o gerenciamento de dados referentes às diversas tarefas que a organização de um evento de porte internacional necessita. Informações sobre estes e outros softwares estarão disponíveis no Stand da WebYES!, na mostra de soluções.

Tamanduá Network Intrusion Detection System

Ernani Azevedo
azevedo@tchelinux.com.br
SLK Linux Group

Técnicas Avançadas de Varreduras baseadas em exploração sutis da pilha tcp

Sandro Melo
sandro@4linux.com.br
4 Linux

Essa palestra tem por objetivo mostrar de forma transparente como os invasores atuam, objetivando informar aos administradores sua metodologia para que os mesmo possam elaboram suas contramedidas. Explorando tecnicamente exploração da pilha TCP como fingerprint, scanners de serviços (TCP, UPD, RPC, SMB) e banners. Destaque para ferramentas de Código aberto: Nmap, Queso, Telnetfp, Nessus.

Tiny Cobol

Rildo Pragana
rildo@pragana.net
Tiny Cobol

O projeto TinyCOBOL é um compilador COBOL que está sendo desenvolvido por membros da comunidade de software livre. A longo prazo, o objetivo é desenvolver um compilador COBOL compatível com o padrão COBOL 85, ou, pelo menos o mais próximo possível.

Torre de Babel: Como Projetos de Software Livre realmente funcionam?

Christian Reis
kiko@async.com.br
Async

Embora existam atualmente milhares de projetos de software publicamente conhecidos, pouquíssimos projetos são conhecidos publicamente; os projetos famosos constituem uma amostra muito pequena e peculiar do que de fato é um projeto de software livre. A realidade é que a `comunidade de projetos' representa uma mistura bastante heterogênea, cada projeto com seus objetivos e políticas próprios.

Esta apresentação utiliza como base um levantamento prático feito entre mais de 500 projetos diferentes, com aplicações e tamanhos diversos; inclui gráficos, estatísticas e comentários inéditos descrevendo este conjunto. Discute assuntos como tamanho de equipe, idade do projeto, garantia de qualidade em software livre, documentação, ferramentas e outros.

Tutorial de J2EE, JDBC

Kemmel da Silva Scopim
kemmel@gci.com.br
UFPR - Universidade Federal do Paraná

Neste tutorial será dada uma visão geral sobre a tecnologia Java e será demonstrado a construção de uma aplicação para WEB acessando qualquer banco de dados. O objetivo do tutorial é demonstrar para pessoas com pouca experiência em Java (ou com nenhuma experiência) flexibilidade de Java para a construção de aplicações para WEB que necessitam interagir com banco de dados relacionais. Será um tutorial para iniciantes em Java.

TV Digital Brasileira e Software Livre

Sérgio Amadeu
sergio.amadeu@planalto.gov.br
ITI
Gilson Schwartz
gschwartz@folhasp.com.br
Cidade do Conhecimento
Israel Bayama
FALTA@EMAIL
Eletronorte

Uma Visão Critica do Programa de "Abertura" de Código da Microsoft Para Governos

Evandro Oliveira
evandro@pbh.gov.br
PRODABEL

Visão Panorâmica do Zope

Palestrante
Luciano Ramalho
luciano@hiper.com.br
Hiperlógica

O objetivo desta palestra é mostrar para este público externo o que é o Zope hoje, para quê ele serve, casos, como funciona a comunidade, panorâmica dos Products existentes, onde o CMF e o Plone se encaixam, e o que esperar do Zope 3.

Você Ainda Usa Banco de Dados?

Klaus Wuestefeld
klaus@objective.com.br
Objetive Solutions

Ridiculamente simples, porém extremamente poderoso, o conceito de prevalência de objetos é o tema desta apresentação: persistência e replicação transparentes para objetos Java comuns usando qualquer máquina virtual Java.

O Prevayler (LGPL), a camada de prevalência original para Java, será demonstrada ao vivo e descrita em detalhe.

Ao final desta apresentação, você será capaz de:

- escrever classes Java comuns e usar objetos normalmente em memória, imune a quedas do sistema;
- queimar seu diagrama Entidade-Relacionamento, deletar sua camada de mapeamento objeto-relacional, jogar fora seus scripts de criação de base e purgar seu código de todo SQL;
- usar qualquer estrutura de dados, qualquer algoritmo e qualquer linguagem de consulta de objetos disponível para Java, nunca mais ficando restrito àqueles fornecidos por seu banco de dados;
- executar consultas em seus objetos de MIL a DEZ MIL VEZES mais rápido que usando um banco de dados via JDBC.

Ou seja, quando você tiver memória RAM suficiente para conter todos seus objetos, usando apenas 350 linhas de software livre, você será capaz de tornar obsoleto o seu banco de dados e estará finalmente livre para usar orientação a objetos de verdade.

URL: http://www.prevayler.org

Workflow em Java: Isso É Bossa, Isso É Muito Natural

Gustavo Vieira
gustavo.vieira@dcc.unicamp.br
UNICAMP

Bossa é uma máquina de workflow em Java, simples, rápida e livre. Nesta palestra vou apresentar a abordagem simples e poderosa do Bossa para workflow, que abre a possibilidade de workflow em todos os lugares.

Um estudo para adoção de software livre na UCS

Edgar Meneghetti, Heitor Strogulski, Alex Pellin, Carlos Vasata

[1]Departamento de Informática
Centro de Ciências Exatas e Tecnologia
Universidade de Caxias do Sul
Rua Francisco Getúlio Vargas 1130, B. Petrópolis
CEP 95070-560, Caxias do Sul, RS

eamenegh@ucs.br, hstrogul@ucs.br, apellin@ucs.br, cevasata@ucs.br

Resumo. *Este artigo descreve o trabalho que está sendo desenvolvido no projeto de migração de uma plataforma baseada em softwares proprietários para uma plataforma livre. Alguns problemas detectados são discutidos assim como algumas soluções de software livre são descritas. São apresentados resultados mais detalhados relacionados a testes realizados com o sistema de correio eletrônico.*

Abstract. *This paper describes the work being developed in order to migrate from a plataform based on a proprietary software into a new one based mostly on free software. Some ploblems detected so far are discussed as well as free software solutions are described. Several tests done on a email system are presented.*

1. Introdução

A Universidade de Caxias do Sul conta com um parque de aproximadamente 2500 microcomputadores e tem usado a plataforma Novell em 30 servidores para uma base de 25.000 usuários distribuídos geograficamente entre vários campus. Em meados de 2002 iniciou-se um projeto que visa migrar deste modelo para uma plataforma que utilize software livre. O projeto tem como objetivo a redução de custo de propriedade de software e a criação de uma infra-estrutura de serviços de rede que permita através de um único login de usuário o acesso a diversos serviços, tais como: sistemas administrativos, biblioteca virtual, portal da internet, quiosque de informações, etc.

O projeto contempla a adoção de software livre em vários serviços essenciais, tais como o sistema de correio eletrônico, sistema de arquivos, intranet, impressão, segurança e monitoramento. Na atual fase do projeto, não pretende-se alterar o sistema operacional dos usuários (GNU Linux, MacOS e Microsoft). A maior incidência de sistemas operacionais corresponde ao Microsoft Windows 98.

A solução para o serviço de diretório e correio eletrônico foi bastante estudada, sendo apresentados alguns resultados neste trabalho. Os demais serviços ainda estão em estudo.

2. A plataforma em estudo

Como mencionado anteriormente, optou-se em manter os sistemas operacionais já existentes nos computadores dos usuários, mais notadamente Microsoft. Desta forma, a opção por uma rede Microsoft pareceu ser o caminho mais adequado, visto que existe um bom suporte para integração entre os protocolos envolvidos neste tipo de rede (CIFS - *Common Internet Filesystem* [4]) e sistemas operacionais abertos (SAMBA [7]).

Partindo desta opção, algumas considerações foram tecidas:

- migração de senhas: as senhas dos usuários estão armazenadas no sistema de diretórios da Novell (NDS) usando algoritmo próprio e de caminho único. Adotando uma rede Microsoft, com autenticação que não transmita senhas abertas pela rede, não se conseguirá utilizar as senhas existentes;
- administração: a administração da rede Novell é bem conhecida e fácil. As permissões do Netware são armazenadas no NDS e estão totalmente integradas a este. Ao migrar o sistema de arquivos para GNU Linux, os tipos de permissões são mais simples e não estão integradas com algum sistema de diretórios, tornando esta opção mais difícil à equipe de administração;
- redundância: a solução em uso é redundante e, de certa forma, tolerante a falhas. Deseja-se manter um mínimo de redundância no sentido de caminhar rumo a uma solução de alta disponibilidade

A pesquisa por soluções adotando software livre levou em conta a necessidade de construir uma infra-estrutura de serviços que pudesse substituir a estrutura atual. As soluções escolhidas até o momento estão listadas nas seções que seguem.

2.1. Sistema de diretório

A necessidade de um sistema de diretório ficou clara desde o início deste trabalho. As duas opções consideradas são a evolução do NDS da própria Novell, porém com suporte a LDAP; e o OpenLDAP [6]. LDAP (*Lightweight Directory Access Protocol*) é um protocolo de acesso a diretório, mais especificamente um serviço de diretório X.500. Embora um serviço de diretório possa armazenar qualquer tipo de informação, observa-se a sua adoção em grandes redes como forma de armazenar informações relativas aos usuários, tais como nomes de usuários e senhas.

2.1.1. eDirectory

O Novell eDirectory [3] é uma evolução do NDS (antigo sistema de diretório da Novell) porém agora com habilitação para LDAP e independência de plataforma. O eDirectory, além da Novell, está disponível para ser executado sobre Windows 2000, Windows NT, Solaris, Linux; podendo operar de forma a integrar estas diferentes plataformas.

No estágio atual, o uso do eDirectory traria a possibilidade de utilizar o aplicativo de gerência de senhas (security login da Novell). Este aplicativo implementa o conceito de *single sign-on*. Porém, a integração do eDirectory com as demais soluções de software livre ainda não estão bem compreendidas pela equipe do projeto.

2.1.2. OpenLdap

O OpenLDAP é um servidor de diretório X.500 leve, além de implementar o próprio protocolo de acesso ao diretório. A forma pela qual as informações do diretório vão ser armazenadas é bastante flexível, incluindo bases de dados mais simples como a biblioteca GDB e BDB [10] ou bancos de dados do tipo MySQL e Oracle. Provê replicação do diretório, além de comunicação segura.

Em testes preliminares, utilizando a base de dados BDB, obteve-se tempos de resposta a consultas bastante satisfatórios quando em teste em conjunto com o sistema de correio eletrônico (comentados na seção 2.2). Cabe salientar que todos os testes foram realizados para uma base de 25.000 usuários.

A possibilidade de utilizar um banco de dados relacional abre possibilidades interessantes do ponto de vista de centralização das informações dos usuários em uma base apenas. A UCS

possui uma base única de alunos, professores e funcionários que é utilizada por vários sistemas para fins diversos, incluindo autenticação. Porém, os serviços de rede não utilizam esta base. Como resultado, tem-se usuários e senhas diferentes para cada tipo de aplicação ou serviço. A centralização destas informações seria altamente benéfica, abrindo a possibilidade para a criação e exclusão de usuários conforme a matrícula e vestibular, por exemplo.

2.2. Sistema de correio eletrônico

Em termos de soluções para correio eletrônico utilizando software livre, pode-se elencar três opções utilizadas pela comunidade: Sendmail, Qmail e Postfix [5]. Desconsiderando comparações de mérito entre eles, optamos por prosseguir utilizando o Postfix, que já é utilizado como *mail relay* da rede. Segundo o próprio autor do software, o Postfix pretende ser rápido, fácil de administrar, seguro e, ao mesmo tempo, ser compatível com o sendmail. Estas características tem levado vários administradores de rede a adotá-lo. Como solução para POP e IMAP, buscamos a implementação da Universidade Carnegie Mellon denominada Cyrus Imap Server [8]. A opção levou em consideração a larga utilização deste pacote pela comunidade e a boa integração com o Postfix. A possibilidade de armazenar as caixas postais no diretório do próprio usuário também foi considerada como altamente desejável, na medida em que possibilita a adoção de limites únicos para o uso de disco.

Em testes de desempenho, o Postfix em conjunto com OpenLDAP apresentou resultados satisfatórios. A integração com um sistema anti-virus é relativamente simples se utilizarmos o software AMAVIS [1] em conjunto com o anti-virus da Network Associates (McAfee) [2]. O Amavis é capaz de extrair os anexos das mensagens (decodificação do MIME) ou deixar a cargo do anti-virus. Embora a segunda opção seja menos confiável, o desempenho é melhor. Todos os testes realizados assumem a existência de um software no servidor para detecção de vírus, de forma que não foram feitos testes com ausência de algum anti-virus.

Algumas experiências também foram feitas com um software para detecção de SPAM (SpamAssassin [11]) com bons resultados na detecção de SPAM, embora com degradação notável de desempenho. A falta de um porte do SpamAssassin para o português ocasiona falhas na detecção de certos tipos de SPAM, principalmente os gerados em língua portuguesa. Isto se dá pela forma adotada na detecção, que atribui pesos a certas características observadas nas mensagens, tais como determinadas *strings* que no software estão descritas em inglês. Para uma melhor identificação de SPAM, seria necessária uma tradução destas *strings*.

O cenário de teste está na figura 1. Para gerar o tráfego SMTP e POP foi utilizado o software Postal e Rabid [9]. A figura reflete o teste completo, incluindo SMTP e POP simultâneamente (item "e"). Para os testes envolvendo MIME foram utilizadas mensagens com anexo contendo vírus. Os resultados (em valores aproximados) são apresentados a seguir.

a) SMTP com anti-virus e Amavis decodificando MIME: 400 mensagens por minuto
b) SMTP com anti-virus decodificando MIME: 500 mensagens por minuto
c) SMTP com anti-virus decodificando MIME e anti-spam: 200 mensagens por minuto
d) POP: 1050 mensagens por minuto
e) SMTP com anti-virus decodificando MIME, sem anti-spam e POP: 500 mensagens por minuto para SMTP e 140 para POP

Observou-se ainda que o sistema é intensivo em termos de uso do processador, principalmente os programas relacionados com a detecção de virus e o serviço de diretório. O uso de memória é baixo e bastante relacionado ao tamanho da cache configurado no OpenLDAP.

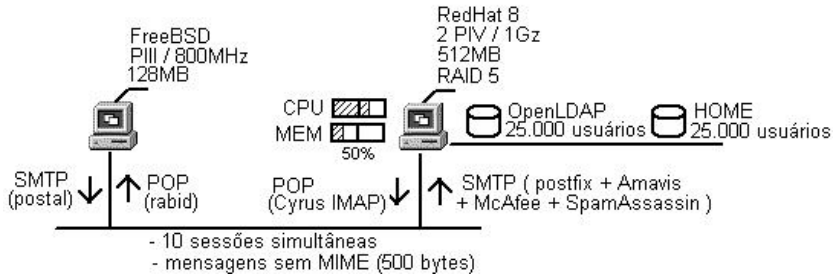


Figura 1: Teste de desempenho do sistema de correio eletrônico

3. Considerações finais

O projeto de migração para software livre implica em modificar boa parte da infra-estrutura de servidores existentes na UCS e portanto apresenta uma complexidade considerável. O porte da rede existente também é um fator que está sendo considerado, principalmente ao avaliar soluções de software livre que nem sempre foram colocados em produção em um ambiente semelhante ao da universidade. Em que pesem as dificuldades encontradas, a solução com software livre supera as mesmas, proporcionando maior flexibilidade e potencialidade para os projetos de TI desenvolvidos na instituição.

Alguns itens ainda merecem um maior estudo e pesquisa, tais como: a conveniência da adoção de uma rede Microsoft, como administrar o sistema de impressão, sistema de arquivos distribuídos, escalabilidade de todos os serviços e alta disponibilidade.

Referências

[1] Amavisd, 2003. Disponível em: http://www.ijs.si/software/amavisd/. Acesso em: mai. 2003.

[2] Network Associates McAfee antivirus, 2003. Disponível em: http://www.nai.com/. Acesso em: mai. 2003.

[3] Novell eDirectory, 2003. Disponível em: http://www.novell.com/pt-br/products/edirectory/. Acesso em: mai. 2003.

[4] CIFS Common Internet Filesystem, 2003. Disponível em: http://www.ubiqx.org/cifs/. Acesso em: mai. 2003.

[5] Postfix, 2003. Disponível em: http://www.postfix.org. Acesso em: mai. 2003.

[6] OpenLDAP Open Lightweight Directory Access Protocol, 2003. Disponível em: http://www.openldap.org. Acesso em: mai. 2003.

[7] Samba, 2003. Disponível em: http://www.samba.org. Acesso em: mai. 2003.

[8] Cyrus Imap Server, 2003. Disponível em: http://asg.web.cmu.edu/cyrus/imapd/. Acesso em: mai. 2003.

[9] Postal SMTP and POP benchmark, 2003. Disponível em: http://www.coker.com.au/postal/. Acesso em: mai. 2003.

[10] Berkeley DB SleepyCat Software, 2003. Disponível em: http://www.sleepycat.com/. Acesso em: mai. 2003.

[11] SpamAssassin, 2003. Disponível em: http://spamassassin.org/. Acesso em: mai. 2003.

Consolidação de Servidores e Serviços para a Comunidade Administrativa e Acadêmica da UNIJUI usando Linux em Mainframe IBM/s390

Agner Quincozes Olson[1], Paulo Sérgio Sausen[2], Sérgio Luis Dill[2]

Universidade Regional do Noroeste do Estado do RS - UNIJUÍ
Departamento de Tecnologia – DeTEC – [2]Informática,
[1]Acadêmico do Curso de Informática
Rua São Francisco, 501 – 98.700-000 – Ijuí – RS - Brasil.

{agner, dill, sausen}@unijui.tche.br

Abstract. *This paper briefly describes the way found by the TI department of UNIJUI to consolidate servers, before, i386 and VM, currently, virtual machines running Linux in IBM S390 mainframe. This servers include administrative and academic scope of the University, which, promote the integration of the various campi through a set of Database Servers, LDAP Servers, E-mail Servers and Web Servers.*

Resumo. *Este trabalho descreve de forma sucinta a maneira encontrada pela Coordenadoria de Informática da UNIJUI para consolidar servidores, antes, i386 e VM, atualmente, em máquinas virtuais (guests) Linux em Mainframe IBM/s390. Servidores estes de abrangência nas áreas Administrativa e Acadêmica da Universidade, os quais, promovem diretamente processos de integração dos diversos campi, através de um conjunto de servidores, de banco de dados, servidor de diretório LDAP, E-mail e servidores Web.*

1. Introdução

A UNIJUI – Universidade do Noroeste do Estado do Rio Grande do Sul é uma Universidade comunitária de cunho filantrópico que abrange mais de cinqüenta municípios do Noroeste do Estado do Rio Grande do Sul. Seu parque de máquinas conta atualmente, incluindo todos os campi, aproximadamente mil computadores PC (Personal Computer), dois computadores Mainframe IBM, dez servidores com aplicações não Linux e uma estrutura de rede de mais de dois mil pontos.

A representação percentual, da distribuição do volume de equipamentos, reflete uma preocupação maior em aparelhar de forma adequada à unidade acadêmica da Instituição, já que mais de sessenta porcento do parque de máquinas é destinado às atividades pedagógicas, sendo utilizados em laboratórios.

Os laboratórios, que estão diretamente à disposição dos alunos dos cursos que pertencem ao Departamento de Tecnologia - DETEC, Engenharias e Informática, possuem mais de um Sistema Operacional incluindo distribuições Linux e FreeBSD. Vale ressaltar que estes laboratórios também atendem alunos de outros cursos da Universidade.

O setor responsável pela manutenção dos serviços de informática prestados pela Universidade é a Coordenadoria de Informática. Esta, pensando na otimização dos serviços e na redução dos custos de manutenção, de suporte às variadas plataformas existentes, e suporte da base de dados dos usuários, adquiriu um equipamento Mainframe IBM z800 Séries com poder de processamento de duzentos e vinte MIPS (Milhões de Instruções por Minuto), oito Gigabyte de memória RAM, uma unidade de disco RAMAC 9393(Virtual Array Storage) com setecentos

gigabytes de memória. No presente a maior concentração das atividades da Coordenadoria de Informática está relacionada à implantação de um novo Sistema de Informação e na migração da base de dados legada, a qual, utiliza Banco de Dados DB2 para VM (Virtual Machine), em produção, paralelamente com a nova base, DB2 para Linux [1]. Para tanto, utilizamos a capacidade de criação de servidores virtuais, hospedados no Mainframe, usando uma distribuição SuSE 7.2 SLES. Criamos também várias imagens desta distribuição, para podermos distribuir a carga dos servidores, entre a base de dados que deverá ser à de produção e as bases que estão sendo utilizadas para testes. [2]

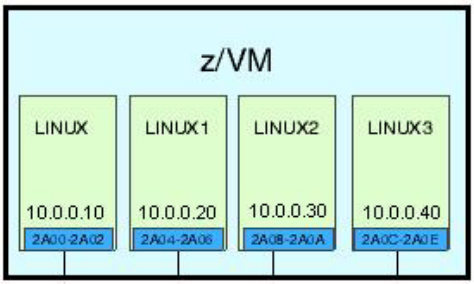


Figura 1 – Imagens Linux (guests) z/VM

Muito dos sistemas que anteriormente eram utilizados em plataformas Intel, atualmente estão em produção em ambiente Linux s390. Este trabalho, que ainda está em fase de conclusão, nos levou a substituir mais de setenta porcento dos servidores PC por diferentes distribuições Linux como máquina virtual em ambiente z/VM. Essa consolidação gera economia, na medida em que não há necessidade de manutenção de hardware para diferentes configurações de equipamentos PC e, no caso dos Sistemas Operacionais, manutenção das licenças e contratos [3].

2. Descrição

No presente momento, estamos em processo de substituição da base de aplicações que anteriormente, era totalmente executada em CSP/VM para novos sistemas que exigem novas plataformas. Uma das soluções encontrada e adequada para as nossas necessidades foi a aquisição de um Mainframe IBM z800 Séries e uma unidade de disco RAMAC Virtual Array Storage 9393. A partir desta aquisição e da aquisição do SIE (Sistema de Informação Educacional), iniciamos o processo de migração, implantação e confecção de sistemas independentes baseados em ferramentas de licença pública.

A escolha destes equipamentos deu-se pela necessidade de mantermos nossa base legada em produção por tempo indeterminado e substituir nosso antigo SGBD por um mais maleável. A escolha do Linux para ser à base de sustentação das aplicações e dos servidores foi devido a sua robustez, segurança, tipos de licença e facilidade de gerenciamento quando instalado como máquina virtual sob zVM.

A utilização dos nossos servidores de Banco de Dados estende-se para além das fronteiras do Sistema de Informação [4]. A partir do momento em que entraram em atividade os servidores SGBD/DB2/Linux, houve uma forte corrente de desenvolvimento de pequenas aplicações, que visaram suprir de forma rápida e eficiente as lacunas causadas pela desaceleração do uso do sistema legado e a implantação do novo sistema. Estas pequenas aplicações, hospedadas em máquinas virtuais, algumas independentes em relação ao desempenho e gerenciamento, envolvem diariamente dezenas de centenas de consultas, tanto ao antigo SGBD como ao atual.

Uma descrição destas aplicações se faz necessário para melhor compreensão da abrangência do escopo deste trabalho:

SGL - Sistema de Gerenciamento de Ligações. Este sistema, desenvolvido com tecnologia JAVA, busca informações armazenadas em um servidor da Central Telefônica da Universidade e transfere os dados coletados para um banco de dados. Neste local estas informações são tratadas e apresentadas aos usuários através de uma interface WEB, onde cada usuário, mediante um cadastro prévio, acessa sua conta, contabiliza suas ligações e envia seus relatórios de uso do telefone, mensalmente.

SCCM - Sistema de Consulta ao Contrato de Mutuo. Este sistema foi desenvolvido para substituir uma antiga aplicação CSP que informava o extrato de conta de mútuo em "terminais burros" distribuídos pela Universidade. Atualmente, esta aplicação busca informações em mais de um servidor e disponibiliza estas informações em páginas WEB utilizando o protocolo HTTPS onde é possível fazer consultas retroativas, agendar saques e fazer pedidos de depósitos para a Rede Bancária Privada.

IPD - Instituto de Políticas Públicas e Desenvolvimento Regional. A partir da reestruturação de um antigo sistema denominado SIB (Sistema de Informações Básicas) foi desenvolvido, utilizando tecnologia JAVA, um novo sistema dinâmico de consulta e inclusão em Banco de Dados. Dados estes, referentes às informações sobre o poder executivo, características geográficas e demográficas, finanças públicas, agropecuárias, industria e comércio, saúde e educação entre outras características de cada município do Estado do Rio Grande do Sul. Estas consultas podem ser feitas dinamicamente na página, e o pesquisador pode montar grupos de dados, grupos de regiões e visualizar graficamente as informações. Este sistema, anteriormente configurado em plataforma Intel, passou ao final de 2002 a ocupar um dos servidores WEB consolidados no Mainframe.

Sistema de Relatório e Balanço Anual: este sistema, desenvolvido em linguagem PHP é utilizado pelos departamentos da Universidade, onde cada usuário através do seu registro geral de funcionário e uma senha, pode acessar o sistema, onde existem diferentes níveis de acesso, para maior segurança. A partir das necessidades dos departamentos, as informações foram agrupadas em cadastros tais como, Apreciativos (parecer geral), Projetos de pesquisa e extensão, Projetos de Ensino, Publicações, Dissertações e Teses, Promoção de Eventos, Participação em Eventos, Laboratórios e Ações Sociais. Para cada cadastro existe a possibilidade de efetuar consultas, incluído aqueles sobre o número de discentes por departamento. O principal objetivo é a geração automática do relatório no final do ano, onde todos os dados são apresentados num formato padrão. A geração pode ser feita tanto em nível departamental ou de forma centralizada, onde todos os dados de todos os departamentos são gerados de uma só vez.

Sistema de Eleições Internas. Desenvolvido com tecnologia PHP, com o objetivo de otimizar os processos de eleições na Instituição, este sistema permite efetuar processos eleitorais sem burocracia. Através de uma senha previamente distribuída, os funcionários e professores exercem o direito de votar sem a necessidade de se deslocar do seu local de trabalho. A segurança do tráfego dos dados na Internet/Intranet é garantida pela utilização do protocolo de comunicação seguro SSL (Secure Sockets Layer).

Um dos projetos de maior impacto nos serviços prestados à comunidade administrativa e acadêmica, ainda em fase experimental, é a criação de um servidor de diretório baseado no protocolo LDAP. Por ser uma Instituição de Ensino, a necessidade da existência de laboratórios de informática, se faz presente em diferentes pontos geográficos, dos quatro campi, que atualmente compõe a UNIJUI e faz-se necessário que cada um dos mais de doze mil alunos, possua endereço eletrônico, login e senha para ter acesso aos referidos laboratórios.

Uma das soluções possíveis, em substituição da anteriormente usada, onde a base de dados dos usuários estava concentrada em ambiente Windows/NT/2000, é a criação de um Servidor de Diretório usando o protocolo LDAP implementado através do software OPENLDAP onde cada

aluno, independente de sua localização, campus ou laboratório, pode usufruir integralmente dos serviços oferecidos à comunidade acadêmica [5].

Algumas das vantagens que este tipo de sistema apresenta: facilidade de manutenção do cadastro dos usuários, e a facilidade como este usuário pode ter acesso aos seus arquivos, lista de contados, e-mail e perfil, independente de sua localização. Outra vantagem apresentada, especificamente no caso da Universidade em questão é que os usuários administrativos, que anteriormente não possuíam login válidos para os diversos laboratórios, agora podem fazê-lo sem restrições.

Outros servidores que fazem parte da estrutura de rede e da estratégia da Universidade para a divulgação de sua marca e de informações à comunidade, bem como inscrições para concursos e vestibulares estão sendo consolidadas de forma gradativa, devido à complexidade e disponibilidade dos serviços.

Entre estes servidores, podemos citar o servidor WEB, portal da Universidade, por ora disponibilizado em plataforma Intel usando FreeBSD, um servidor Webmail, um servidor Proxy usando o software squid e um servidor Firewall ainda em fase de implantação.

3. Conclusão

Este trabalho é um projeto inovador, que por seu pioneirismo, tem esbarrado na escassez de referenciais sólidos, mas por sua vez desafiador.

A forma de apresentação e implementação das atividades descritas sugere um grande investimento inicial para que no decorrer das ações mostre-se viável, reduzindo custos desnecessários especificamente no tocante aos custos de licenças proprietárias.

As soluções encontradas durante as fases de projeto e implementação mostraram-se bastante maleáveis, seguras e eficientes. Devido à natureza experimental de algumas construções, acreditamos em algum momento haver a necessidade de mudança nos esboços dos projetos, o que não invalidaria as ações implantadas até o momento.

De forma geral todos os sistemas consolidados e instalados ganharam em segurança, robustez, desempenho e facilidade de manutenção justamente por estarem concentradas em um local e serem passíveis de consulta e gerenciamento de qualquer ponto da estrutura de rede da Instituição.

Em trabalhos futuros temos previsão de que todos os serviços vitais para a Instituição, outrora suportados por plataformas de pequeno porte, passem a ser usadas completamente dentro da estrutura de máquinas virtuais (guests) Linux em Mainframe.

4. Referências Bibliográficas

[1] Date, C.J. **Introdução a Sistemas de Bancos de Dados** (tradução da 4ª ed.). Rio de Janeiro: Editora Campus, 1991.

[2] Geiselhard, Gregory; Chong, Tung-Sing; Donovan Michael; **Linux on IBM @server zSeries and S/390: Cloning Linux Images in z/VM**. 2002.

[3] AMREHN, Erich; Jordan, Joachin; Kirschner, Frank; Reeder, Bill; **Server Consolidation with Linux for zSeries.** 2002.

[4] SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistemas de Bancos de Dados**. 3a Edicão. Makron Books, 1999.

[5] AMREHN, Erich; BOCHE, Ulrich;GNIRSS, Dr. Manfred: **Securing Linux for zSeries with a Central z/OS (RACF) LDAP Server**. 2002.

Proposta de uma Política de Software Livre em Instituições de Ensino Superior: Um Estudo de Caso

Francisco J. Monaco[1], **Celso R. G. Providelo**[2], **Patricia Bilotta**[2]

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo
Av. Trabalhador São-Carlense, 400 - Centro 13560-970 São Carlos, SP

[2]Departamento de Engenharia Elétrica - EESC — Universidade de São Paulo
Av. Trabalhador Sãocarlense, 400 - Centro 13566-590 São Carlos, SP

[3]Departamento de Hidráulica e Saneamento - EESC — Universidade de São Paulo
Av. Trabalhador Sãocarlense, 400 - Centro 13560-250 São Carlos, SP

monaco@icmc.usp.br, cprov@sel.eesc.usp.br, pbilotta@sc.usp.br

Abstract. *Either in virtue of technological quality or socioeconomic suitability, the preferential adoption of free software gains importance in the resource policy of a growing number of public and private institutions, where it has been established as an alternative to the monopolies of the proprietary industry and assumes relevance in governmental strategy for development and national security. It is no exception the increasing interest in infrastructure updating projects aiming at the adoption of the GNU/Linux platform in both academic and administrative activities. This paper introduces a case study of a supporting program for free software adoption in an education and research center and a summary of its goals, which are presented as suggestion of subsidiary arguments for related initiatives engaged in the conception of similar proposals.*

Resumo. *Seja em virtude da qualidade tecnológica ou adequação sócio-econômica, a opção pela adoção prioritária de software livre vem ganhando importância na política de recursos de um número cada vez maior de instituições públicas e privadas, onde estabelece-se como alternativa aos monopólios da indústria proprietária e assume relevância nas estratégias governamentais de desenvolvimento e segurança nacional. Não constitui exceção o crescente interesse em projetos de atualização da infra-estrutura do parque computacional em instituições de ensino visando a adoção preponderante da plataforma GNU/Linux em atividades acadêmicas e administrativas. Este artigo introduz um estudo de caso de um programa de suporte à adoção de software livre em um centro de ensino e pesquisa e uma síntese de seus objetivos, os quais são apresentados como sugestão para argumentos subsidiários à iniciativas correlatas empenhadas na concepção de propostas semelhantes.*

1. Introdução

O programa-piloto descrito neste artigo, apresentado à Comissão de Informática do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, foi concebido em atendimento à decisão institucional daquela Unidade em favor de um plano para implantação do sistema GNU/Linux de forma predominante na infra-estrutura do seu parque computacional. Subsidiado pela experiência anterior angariada através das iniciativas do projeto GNUsp[1] [Rodrigues et al., 2001, Monaco et al., 2002, Monaco, 2002] no campus universitário, a

[1]O Projeto GNUsp (`www.gnusp.org`) é uma iniciativa científico-educacional liderada por professores, estudantes e contribuidores voluntários cujo objetivo é promover o conhecimento acerca de fundamentos conceituais e técnicos

proposta elaborada para a implementação do projeto ressalta que, não obstante o destacado papel que a plataforma GNU/Linux tem desempenhado como instrumento viabilizador de ações nesse sentido, os benefícios almejados mediante sua introdução são pouco efetivos se estratégia equivalente não conduzir a seleção de aplicações oferecidas aos usuários, i.e., se estes não dispuserem também de aplicativos e serviços (editores de texto, bancos de dados, ambientes de desenvolvimento etc.) não proprietários à sua disposição.

Argumenta-se, por conseguinte, que o propósito da recomendação para a implantação do referido sistema operacional resulta em maior alcance se definido como meta dentro de um programa mais amplo de *suporte à adoção de software Livre*, o qual contemple não somente a disponibilização do SO nas estações de trabalho, mas também a disseminação de aplicações livres, incluindo utilitários tradicionais e sistemas servidores, bem como a orientação para a utilização de padrões não proprietários para armazenamento e transmissão de dados tais como formatos de arquivos empregados no arquivamento e divulgação de informações, incluindo correio eletrônico, páginas Web e documentos administrativos. Os objetivos e plano de execução do referido programa são o assunto das seções seguintes.

2. Objetivos

O Programa de Suporte à Adoção de Software Livre tem como objetivo, além da assistência ao processo de implantação da plataforma GNU/Linux, oferecer apoio técnico à migração das aplicações em utilização para o novo sistema, proporcionar orientação para seleção e avaliação de soluções baseadas em SL para aplicações específicas e estimular a concepção de iniciativas práticas nesse terreno. Como metas gerais para a primeira fase do referido programa, o seguinte plano de ações é sugerido em prol dos objetivo assinalados.

a) **Adoção de soluções baseadas em SL em sistemas e serviços institucionais.**

   Dirimindo custos de licenciamento de produtos proprietários, a adoção de uma política de Informática baseada em SL atende não apenas à desoneração de investimentos em software, mas a uma maior eqüidade nas condições de acesso à tecnologia de informação, proporcionando efetiva medida mitigadora ao alcance das ações de combate à exclusão digital.

   Ao prescindir de soluções fechadas de utilidade genérica em favor de sistemas flexíveis de funcionalidade extensível e personalizável, a priorização do SL como infra-estrutura do parque computacional, além da gestão mais eficiente dos recursos de informática, representa uma decisão estratégica para a manutenção da autonomia institucional.

b) **Exploração do SL como recurso didático e sua integração em atividades acadêmicas.**

   Além de ferramentas de utilidade direta, artefatos de SL constituem inerentemente instrumentos didáticos em si, garantindo acesso à toda informação disponível em seu código-fonte e documentação associada. Assim, mais do que treinar-se na sua utilização, é possível estudá-lo, adaptá-lo e aperfeiçoá-lo, propiciando valioso recurso para o aprendizado.

   A integração de ferramentas baseadas em SL nas disciplinas regulares de graduação e pós-graduação, ao eliminar custos de licenciamento de alternativas proprietárias, além de eliminar impedimentos financeiros à adoção do estado-da-arte em tecnologia de software, permite ao

em Software Livre entre usuários e desenvolvedores profissionais. Desde seu início, o Projeto tem suportado diversas atividades no campus universitário, incluindo a organização de cursos e seminários e manutenção de recursos para suporte à comunidade de usuários. O projeto HackAdemia [Monaco et al., 2003] e a elaboração do programa institucional de adoção de software livre, tema deste trabalho, são dois exemplos de atividades atualmente em desenvolvimento. Planos para equipar um Laboratório de Software Livre dedicado ao desenvolvimento de pesquisa acadêmica na área encontram-se na lista de metas para empreendimentos futuros.

aluno a oportunidade de explorar mais efetivamente seu objeto de estudo, inclusive mediante a possibilidade de obtê-lo gratuitamente para uso pessoal (p.ex. em seu domicílio para tarefas complementares, ou em programas de ensino a distância) ou profissional. Adicione-se a essas a vantagem da liberdade para distribuir publicamente artefatos produzidos através de atividades didáticas ou científicas (a qual pode não é plena sob o impositivo legal de licenças de desenvolvimento restritivas) e o alcance de tais produtos a toda sociedade (que deles pode desfrutar livremente).

c) **Estímulo ao exercício do aprendizado através da prática em projetos de SL**

O notório entusiasmo com que estudantes dedicam-se a projetos de SL fundamenta-se na experiência de trabalhar em conexão com sua comunidade profissional e o gratificante reconhecimento, por parte daquela, de seu mérito em contribuir com uma empreitada de sucesso. A exploração desse potencial na prática didática propicia não somente estímulo para o aprendizado, mas também oportunidade para integrar conhecimento, desenvolver auto-confiança e exercitar talentos, quer habilidades técnicas, quer caracteres de liderança.

Em adição à contribuição voluntária — seja como iniciativa pessoal, seja no âmbito educacional — o envolvimento na comunidade internacional de desenvolvedores como atividade profissional ganha crescente relevância no panorama da indústria de software. Realocando recursos da aquisição de licenças para o desenvolvimento e otimização continuada, a demanda de novos especialistas em computação capacitados para essa emergente realidade torna-se uma nova necessidade a ser contemplada pelos cursos de formação na área.

3. Plano de Execução

A experiência angariada mediante empreendimentos de mesma natureza demonstra que uma das condições essenciais para encorajar a participação da comunidade de usuários no programa de implantação de SL e propiciar que esta transcorra positivamente é o oferecimento de uma infra-estrutura de suporte adequada. O atendimento de tal necessidade inclui a formação de um *núcleo de apoio* capaz de orientar e prestar assistência aos usuários, facilitando a migração para a nova plataforma e propiciando que esta se processe tranqüilamente e com segurança. Dentre outras facilidades a serem proporcionadas dessa forma, algumas são de particular importância:

a) um balcão de orientação onde usuários possam obter informações acerca de disponibilidade e adequação de soluções em software livre para aplicações específicas, instruções para instalação, configuração e utilização de programas, inclusive no que concerne à modalidade de licenciamento e sua compatibilidade com a política de software livre [Stallman, 1995];
b) uma infra-estrutura de comunicação que facilite a troca de experiências e auxílio mútuo entre usuários, especialmente dirigido aos membros do Instituto, mas propiciando a interação com a comunidade extra-universitária engajada no oferecimento de suporte à iniciativas nesse campo;
c) uma central de recursos para projetos de adoção de SL, incluindo base telemática de conhecimento (na Web), repositório de software, serviços de apoio on-line, ferramentas de migração inter-plataformas etc.;
d) um programa de treinamento para usuários e administradores de sistema através de cursos regulares e sob demanda, visando promover a capacitação da comunidade para propor e implementar iniciativas na implantação de software livre, bem como para geri-las e solucionar dificuldades.

O caráter do comprometimento voluntário e espírito de equipe são inerentes à natureza do movimento SL, e considerados pelos seus praticantes como essenciais ao seu desenvolvimento consistente, conforme corroborado pelos casos de sucesso fundados nesse paradigma (cite-se o

próprio histórico do Projeto GNU como exemplo). Por essa razão, a formação de uma equipe de trabalho para organizar o núcleo de suporte ao programa de adoção de software livre deve acolher a iniciativa dos seus proponentes, a qual deve ser encorajada através de seu engajamento direto no projeto. Com referência às resistências que não raro se manifestam contrárias à implementação de programas desse porte e natureza, a estratégia de ação do núcelo de suporte não deve ser a de exercer-lhes pressão, mas a de respeitá-las e, procurando identificar a origem das objeções (receio de mudanças, equívocos técnicos etc.), buscar dissipá-las mediante a disseminação da informação. A meta é angariar adesões espontâneas ao programa despertando o interesse da comunidade de usuários — por exemplo, organizando seminários regulares para apresentar soluções baseadas em SL, seus recursos e casos de sucesso.

4. Conclusões

O estudo de caso desenvolvido neste artigo descreve a concepção de programa de suporte à adoção de software livre em uma instituição de ensino e pesquisa, apresentando os argumentos em favor de sua implantação, objetivos almejados e estratégia de execução. A principal conclusão assinalada é a de que os benefícios esperados mediante tal empreendimento tornam-se mais efetivos se este desenvolver-se sob um programa amplo que objetive não apenas a disponibilização de um sistema operacional livre nas estações de trabalho do parque computacional da instituição, mas também promova a disseminação de SL em aplicações para usuários, incluindo utilitários tradicionais, pacotes didáticos e oriente a utilização de formatos não proprietários de codificação de dados.

Para tal, propõe-se a implantação de um *programa de suporte à adoção de software livre*, mantido por um *grupo de apoio* adequadamente capacitado, dedicado à oferecer assistência à execução desse processo, compreendendo orientação para adequação de soluções em software, instruções de configuração e utilização, bem como compatibilidade e interoperabilidade com sistemas já implantados. Mediante essa facilidade, fundamental para encorajar a participação da comunidade de usuários, o programa busca criar condições não somente para a implantação da plataforma nos serviços institucionais, mas também sua integração nas atividades acadêmicas como recurso didático, seja através do estímulo ao envolvimento dos usuários junto à comunidade de desenvolvedores, seja através da sua integração em disciplinas regulares oferecidas pelo Instituto.

Referências

Monaco, F., Providelo, C. R. G., and Bilotta, P. (2003). Hackademia: : Uma experiência com preparação de novos desenvolvedores para a comunidade software livre. In *Workshop Software Livre 2003, IV International Forum on Free Software*, Porto Alegre, Brazil. Sociedade Brasileira de Computação (SBC).

Monaco, F. J. (2002). Il software libero è educativo: validi motivi per cui l'open source può promuovere l'università virtuale. *Form@are - newsletter per la informazione in rete, Centro Studi Erickson di Trento*.

Monaco, F. J., Providelo, C. R. G., Rodrigues, W. J., and Borges, J. H. G. (2002). Experiences with the concept of free software in a professional education environment. In *Workshop Software Livre 2002 (SBC), III International Forum on Free Software*, pages pp. 78–81, Porto Alegre, Brazil.

Rodrigues, W. J., Monaco, F., Gonzaga, A., and Aguiar, M. L. (2001). Free software in superior education : Talking about freedom to future professionals. In *Workshop Software Livre 2001 (SBC), II International Forum on Free Software*, pages pp. 43–45, Porto Alegre, Brazil. Sociedade Brasileira de Computação (SBC).

Stallman, R. (1995). The free software foundation. http://www.fsf.org.

Software Livre Suportando Ambientes Científicos de Alta Tecnologia

Celso Providelo[1] **, Francisco J. Monaco**[2] **, Ralf Keding**[3]

[1]Departamento de Engenharia Elétrica
Escola de Engenharia de São Carlos - Universidade de São Paulo
Av. Trabalhador Sancarlense, 400 – 13566-530 São Carlos, SP

[2]Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
Av. Trabalhador Sancarlense, 400 – 13560-970 São Carlos, SP

[3]Otto-Schott Intitut für Glaschemie
Friedrich-Schiller Universität
Fraunhoferstr, 6 – 07743 Jena, Deutschland

cprov@sel.eesc.usp.br, monaco@icmc.usp.br, Ralf.Keding@uni-jena.de

Abstract. *This article deals with the advantages of using Free Software and its methodologies in the development of high-end scientifc instrumentation systems. Highlighted advantages are presented not only in the academic context, but also in the industrial environment. The system* OSI-VGA *in development in* Otto-Schott Institut für Glaschemie *in Germany is used as a case study.*

Resumo. *Este artigo trata das vantagens alcançadas ao se adotar software livre e suas metodologias no desenvolvimento e implantação de sistemas de instrumentação científica de alta tecnologia. Vantagens destacadas são apresentadas não apenas no contexto acadêmico, mas também em ambientes industriais. O sistema* OSI-VGA *em desenvolvimento no* Otto-Schott Institut für Glaschemie *na Alemanha. é discutido como estudo de caso.*

1. Introdução

Ao contrário da equivocada primeira impressão que ocorre àqueles que se surpreendem perante o "paradoxal sucesso" do Software Livre [Stallman, 1984] a despeito de seus princípios e métodos não ortodoxos (do ponto de vista das metodologias tradicionais vigentes na indústria proprietária), seus proponentes não creditam seus bons resultados à inteligência peculiar dos membros de sua comunidade, mas ao seu modelo de desenvolvimento aberto e cooperativo, baseado no acesso não privilegiado à informação. Na vanguarda do seu movimento, já se cogita, em diversas instâncias, que a experiência é válida para outras áreas do conhecimento tais como a indústria da música, da imprensa e da eletrônica[1]. Este artigo comenta a influência da disseminação do Software Livre no desenvolvimento de sistemas de Instrumentação Científica e discute seus impactos através de um estudo de caso.

1.1. Sistemas de Instrumentação Científica

Os sistemas de instrumentação científica considerados nesse trabalho compreendem a associação de técnicas, ferramentas e profissionais multidisciplinares com o intuito de elaborar novos conceitos em uma área específica do conhecimento. Mais que a utilização de SL na implantação deste

[1]*Free Music*, *Free Print* e *Free Hardware*, dentre outras propostas, já são largamente discutidas na Internet.

tipo de sistema, este artigo discute mudanças mais profundas no que diz respeito à concepção e constante adaptação destas aplicações.

2. Estudo de caso

Um exemplo de sistemas de instrumentação científica é o *LabScape* [Arnstein L., 2002] desenvolvido na *Universidade de Washington* . Sistemas deste gênero associam ferramentas disponíveis como: base de dados, sistemas automáticos de aquisição de dados, ferramentas computacionais de cálculo, sistemas de diagnóstico assistido por computador, entre outros, para auxiliar o indivíduo em tarefas que requerem alta confiabilidade, tais como: pesquisa científica, manutenção preditiva, controle de processos industriais e agricultura de precisão.

O sistema que apresentamos, o OSI-VGA [Providelo and Panosso, 2001], em desenvolvimento no *Otto-Schott Institut für Glasschemie* na cidade de *Jena* , *Alemanha* é utilizado tanto na análise laboratorial e pesquisa de novos materiais (meio acadêmico) quanto no controle de precisão dos níveis de pureza na linha de produção de vidros planos (meio industrial).

O diagrama da Figura 1 mostra o funcionamento do sistema: requisição de medida (1), gerenciamento de sessão(2), aquisição de dados (3), armazenamento dos dados (4), aviso de encerramento (5)[2]. A arquitetura geral da aplicação é ilustrada na Figura 2.

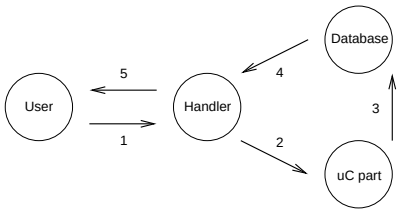


Figura 1: Diagrama do Sistema OSI-VGA

O componente de software aparente ao usuário foi escrito em C++ utilizando a biblioteca gráfica QT2. O desenvolvimento se deu com a ajuda do *Kdevelop* [kde, 2000] um IDE (Integrated Development Environment), o qual facilitou o trabalho de compilação, *debug* e teste. O "handler", ou manipulador de tarefas, foi escrito em *C-ANSI*, visando melhor desempenho e implementa todas as funções destinadas ao compartilhamento dos recursos utilizados, gerencia sessões e mantem a integridade e segurança do sistema.

A parte microcontrolada foi implementada a fim de realizar as tarefas que necessitam de coerência temporal (*Hard Real-Time* [Tanembaum, 2001]), aliviando a carga de processamento nos componentes superiores e evitando *patches* no Kernel Linux atual (que não dispõe de capacidades adequadas para a aplicação). A interface neste ponto ocorre por meio de um duto de mensagens via porta serial utilizando a técnica *HRI*[3] [Providelo, 2001].

A base de dados utilizada foi PostgreSQL [pgs, 2000] pela funcionalidade e confiabilidade oferecida, e o *front-end* do sistema é mostrado na Figura 3, onde vê-se a GUI que auxilia a visualização dos dados e a elaboração do processo de medida, criando uma interface rápida para o trabalho de pesquisa.

[2]Utiliza-se o protocolo *IDNP* para redes de dispositivos interativos [Monaco, 2002]
[3]*Human-readable Interface*.

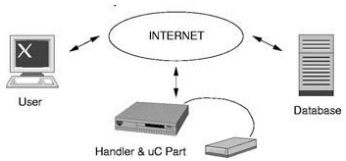


Figura 2: Topologia do Sistema OSI-VGA

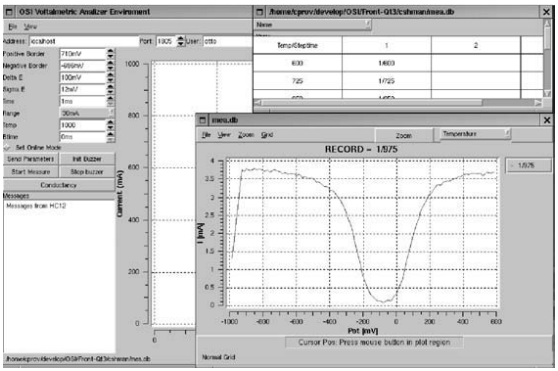


Figura 3: Screenshot do Sistema OSI-VGA

3. Sobre a metodologia utilizada

Como comentado na seção 1, o projeto foi concebido segundo os princípios e técnicas consagrados na Comunidade Software Livre, utilizando a infra-estrutura de uma incubadora de projetos (a popular *Sourceforge*), a qual disponibiliza os serviços de forma gratuita e confiável na Internet. Foi criado um repositório, um *bug-tracker*, uma lista de discussão via e-mail, um *site* para a divulgação de notícias e pedidos de ajuda.

O número de participantes não é nada comparado a projetos nitidamente mais populares como *Apache* ou *Mozilla* — assim como *Eric S. Raymond* sugere em *The Cathedral and the Bazar* [Raymond, 1998] — no entanto o nível de sofisticação real do software gerado é muito alto se comparado outros softwares proprietários disponíveis no mercado. Credita-se tal fato à possibilidade de os usuários, mesmo em número reduzido, terem participação direta no desenvolvimento do sistema. Mesmo aqueles que não são programadores têm acesso à toda informação de que necessitam para entender e opinar no projeto, o que, além de propiciar seu aprendizado mesmo em tópicos alheios ao seu conhecimento específico, constitui orientação valiosa para a equipe de desenvolvedores. É clara a importância de se seguir consistentemente as recomendações que facilitem essa interação, tais como: simplicidade, clareza, boa documentação (no que concerne à programação, em particular, um bom guia é o *GNU Code Standarts*, escrito por Richard M. Stallman e disponibilizado pela *Free-Software Foundation* [Stallman, 1984].

4. Lições e Conclusões

Em comum com outros projetos de mesma natureza, este trabalho ressalta o modo como a adoção de ferramentas utilizadas no processo de desenvolvimento de software livre, no âmbito que se refere ao acesso a informação e respaldo ao trabalho colaborativo, propicia o intercâmbio de

informação entre os envolvidos, acelerando o processo de aprendizado e mitigando deficiências entre componentes do grupo. Outro aspecto importante que deve ser levado em consideração é a satisfação de "ensinar aprendendo", ou seja, o profissional envolvido em um projeto concebido sob os princípios do *Software Livre* associa seus conhecimentos ao conhecimento comum do grupo e o assimila, fazendo com que ele se beneficie do conhecimento dos outros integrantes também.

O que torna o exemplo do sistema OSI-VGA particularmente interessante, todavia, é a forma como sua experiência evidencia a disseminação do SL como desafio à indústria do conhecimento proprietário. Sua influência no projeto em exame não se restringe ao software produzido, mas foi capaz de motivar toda uma equipe de trabalho, incluindo programadores, projetistas de hardware e usuários cientistas a assumirem uma atitude coerente em relação ao sistema completo que, desde a documentação de operação até especificação dos circuitos eletrônicos, está sendo disponibilizado sob a licença GNU GPL.

Tem-se observado vantagens de sistemas deste gênero no ambiente acadêmico-industrial, pois trazem consigo algumas qualidades muito desejáveis não só no software, em seu aspecto técnico, mas no sistema como um todo, qualidades estas provenientes desta nova metodologia de desenvolvimento, não apenas em termos modularidade, facilidades no compartilhamento de recursos e escalabilidade, mas também na flexibilidade e autonomia permitidas pela liberdade de colaborar propiciada pelo modelo SL. Por outro lado, evidenciam-se os conflitos pertinentes ao trabalho científico: sob o ideal de produzir conhecimento e disseminá-lo para o bem comum, o benefício de desfrutar do software livre constitui, em si, um desafio ético à sua utilização na produção de conhecimento proprietário, o que demonstra como o impacto do SL no ambiente científico tende a ir além das meras aplicações computacionais.

Referências

(2000). Kdevelop. http://www.kdevelop.org.

(2000). Postgresql. http://www.postgresql.org.

Arnstein L., Borriello G., C. S. H. C. S. J. (2002). Labscape: A smart environment for the cell biology laboratory. *IEEE Pervassive Systems*, vol. 1(no. 3).

Monaco, F. J. (2002). *Proposta de uma rede para dispositivos telemáticos*. PhD thesis, EESC-USP.

Providelo, C. (2001). Concepts of human readable interface (hri).

Providelo, C. and Panosso, A. (2001). Microcontrolled implementation of a voltametric analizer in glasses. Technical report, Otto-Schott Institüt für Glaschemie and SEL-EESC-USP.

Raymond, E. S. (1998). *The Cathedral and the Bazar*. O'Reilly.

Stallman, R. M. (1984). www.fs.org.

Tanembaum, A. S. (2001). *Modern Operating System*. Prentice Hall, second edition edition.

Implementação de uma rede de computadores para ensino de física

Acquadro, R. V.[1], Gaspar, C. E.[1], Raggio, P. M.[1]

[1]Instituto de Física Gleb Wataghin – Universidade Estadual de Campinas (UNICAMP)
Caixa Postal 6561 – CEP 13084-970 – Campinas – SP – Brasil

racquad@led.unicamp.br, carlosgaspar@yahoo.com, praggio@ifi.unicamp.br

Resumo. *O trabalho realizado mostra a viabilidade de software livre em uma rede de computadores para o ensino de física. Foram usados computadores fadados a sucata para revitalizar uma rede e reaproveitá-la para beneficiar os estudantes do instituto de física. O software contruído concomitantemente com a rede também ilustra as inúmeras vantagens na adoção de software livre.*

Abstract. *The work shows the viability of the free software in a network for the physics teaching. Obsolete computers were used along a new network to benefit the students of the physics institute. The software developed also shows the advantages in the use of free software.*

1. Introdução

Nos últimos anos, o uso de recursos computacionais no ramo da educação tem evoluído de maneira espantosa, tanto na melhoria da qualidade quanto na quantidade de ferramentas. Então, por que não explorar melhor estes recursos, tornando os estudos mais concretos e agradáveis aos estudantes?

Hoje em dia, a computação tem se tornado uma ferramenta imprescindível para o ensino e pesquisa de física, sendo utilizada em aquisições de dados, tratamentos estatísticos, simulações, entre tantas outras aplicações. Mas, como conciliar física e computação em tempos de crise, baixas verbas destinadas a compra de recursos computacionais e altos preços de softwares? Isto é possível através da adoção de softwares de baixo custo, mas principalmente através da adoção de software livre pelas suas inúmeras vantagens (especialmente: custo, facilidade de adaptação/personalização e constante desenvolvimento através da colaboração entre os usuários).

Este trabalho procura mostrar como é possível utilizar software livre em uma rede destinada ao ensino de física. O software simulador desenvolvido e toda a configuração da rede de computadores é um bom exemplo de aplicação computacional de baixo custo voltado para o ensino de física.

É importante ressaltar que durante estre trabalho não foi usado em momento algum qualquer software proprietário. Desde o desenvolvimento do software até a escrita desta publicação, foram usadas apenas ferramentas livres e de fácil aquisição na Internet.

2. Software

2.1. Considerações e objetivos:

O software simulador foi desenvolvido totalmente a partir do zero, mas inspirado em softwares já existentes. O objetivo do software é oferecer ao aluno de física uma ferramenta de simulação de exercícios que podem ser normalmente encontrados nos livros didáticos [1, 2, 3], auxiliando o entendimento da disciplina. O software também pode ser útil ao educador, por oferecer uma maneira simples, rápida e prática de demonstração dos exemplos estudados em sala de aula.

Para atender a estes objetivos, o software se propõe a resolver um certo número de problemas normalmente encontrados nas disciplinas ministradas durante os cursos de mecânica clássica.

2.2. Caractersticas e funcionamento

Em seu atual estágio de desenvolvimento, o software conta com quatro tipos de problemas: queda de corpos, osciladores com molas, lançamento de projéteis e pêndulos, todos, com exceção do módulo de pêndulos, levando em consideração atrito viscoso com o meio. A idéia de modularizar o software torna mais fácil uma possível atualização, aumentando a gama de problemas oferecidos. Existe um corpo principal do software onde cada módulo responsável por uma das simulações é devidamente inserido. O programa principal chama os módulos separadamente, dependendo da escolha do usuário. Desta forma é fácil inserir sub-rotinas para a simulação de outros exercícios de mecânica e até de outras áreas da física.

A linguagem de programação adotada para o desenvolvimento do software foi a linguagem C [4], uma das mais utilizadas junto a comunidade acadêmico/ científica. Isso foi feito pensando que, no futuro, outros usuários possam realizar alterações ou implementações do programa com certa facilidade. Outra vantagem da linguagem C é sua portabilidade para outros sistemas operacionais. Inicialmente, ele foi desenvolvido para Linux, mas poderá ser compilado em qualquer outro sistema operacional que suporte linguagem C padrão ANSI.

O software é todo executado em modo texto, mas para que o usuário possa ver os gráficos gerados, é necessário que ele possua acesso a alguma interface gráfica (seja ele via SVGALib ou via X-Window). Se o usuário não tiver acesso a este tipo de recurso, o software apenas apresentará os resultados numéricos. Mas, mesmo sem acesso a interfaces gráficas, ainda existe a possibilidade de salvar os gráficos em um arquivo de formato PNG ou PostScript. A capacidade de salvar gráficos em arquivos é extremamente útil quando deseja-se, por exemplo, ilustrar um relatório ou uma publicação. O formato PNG é suportado por uma gama grande de editores de texto, livre e não livres.

Para apresentar o resultado gráfico do problema, o software gera um script que é lido pelo GNUPlot[5]. Neste script estão contidas todas as informações necessárias para a construção do gráfico desejado. O software GNUPlot foi escolhido pois é de fácil programação, pode ser encontrado em qualquer distribuição Linux e é livre.

A interface do software é a mais simples possível. As opções são acessadas a partir de menus, onde cada opção chama o módulo respectivo do software, onde o usuário deverá entrar com dados específicos para o problema. Após a entrada dos dados o software se encarrega de apresentar soluções numéricas e gráficas.

2.3. Algorítmos empregados

Em geral, os problemas de mecânica clássica são descritos por equações diferenciais de segunda ordem; porém, em nem todos eles é possível chegar a uma solução analítica. Nesses casos é necessário utilizar métodos numéricos para encontrar as soluções. Dentre os quatro módulos propostos, existiram duas situações em que foram necessários o desenvolvimento de sub-rotinas especiais para a resolução das equações.

Um desses problemas encontrados foi o cálculo do alcance de um projétil com resistência com o ar; este alcance é dado pela raíz de uma função transcedental. O método adotado para este caso foi o método de Newton para encontrar raízes de uma função. A sub-rotina escrita retorna, depois de um número fixo de iterações, o valor do alcance.

Já para o cálculo do período de oscilação do pêndulo, normalmente usa-se a aproximação de sen $\theta \approx \theta$, quando trata-se de ângulos muito pequenos. Foi escrita uma sub-rotina que um utiliza método numérico para calcular o período de oscilação em pêndulos para qualquer elongação. Analíticamente, utiliza-se métodos de integrais elípticas, mas estas podem ser transformadas em polinômios, simples de serem calculados [6].

2.4. Histórico

Desde a versão 0.4 do simulador (lançada no dia 18 de Setembro de 2002) o programa esteve disponível na Internet para que qualquer interessado pudesse baixar o código fonte. Graças a isso, o programa recebeu grande colaboração de Daniel Ome, professor de segundo grau que, mesmo geograficamente distante (morando na Argentina), esteve muito presente e ativo durante o desenvolvimento, colaborando muito com a resolução de bugs de programação. Isso só foi possível porque o software foi licenciado sob os termos da GPL, tornando-o totalmente livre. A partir da versão 1.0 (atual versão disponível), além do código fonte, os arquivos executáveis compilados para Linux também estão disponíveis para download no seguinte endereço: http:// sourceforge.net/projects/mecanica.

Ao término do desenvolvimento, foi escrito um Guia do Usuário, para que os usuários possam ter uma orientação sobre o uso do programa. Este guia de referência explica como o problema é simulado, quais as equações envolvidas nos cálculos e quais os resultados retornados ao usuário pelo software e também pode ser encontrado no mesmo endereço do software.

3. Rede de computadores

Paralelamente com o desenvolvimento do software, uma rede experimental composta por três PCs IBM 486 foi montada, para que pudessem abrigar software úteis para o uso dos estudantes de física. Atualmente, a rede encontra-se montada, mas apenas como demostrativo de como é possível utilizar software livre e micros antigos para uso de alunos de um modo geral, atendendo de maneira satisfatória às suas necessidades.

3.1. Características da rede

Apesar da obsolecência dos PCs 486, foi possível, graças ao software livre, demonstrar como pode-se revigorar e utilizar de maneira satisfatória esse tipo de máquina, fadada ao esquecimento.

Com um pequeno custo de aquisição de um novo computador, é possível concentrar todo o processamento dos aplicativos neste servidor, e distribuir apenas as informações já processadas para todos os clientes. Dessa maneira, é possível ter o desempenho de um micro de ponta em clientes obsoletos.

Inicialmente instalamos alguns softwares nos clientes, como AbiWord, Gnumeric, Galeon, entre outros. Mas com a idéia de um servidor ligado na rede, pode-se instalar outros softwares mais modernos e completos, como Mozilla, Evolution, OpenOffice, entre tantos outros.

4. Resultados alcançados

Apesar da versão final do software ter sido lançada dia 22 de Novembro de 2002, há pouco mais de 6 meses, já foram colhidos alguns resultados muito importantes:

- Participação ativa de um professor de segundo grau argentino no desenvolvimento do programa;
- Criação de um Grupo de Usuários de GNU/Linux da Física (GULF);
- Continuação do projeto por outro aluno de graduação.

5. Projetos para o futuro

- Desenvolvimento de um front-end para tornar o software mais amigável para o usuário. Este front-end poderia ser ainda em modo texto (usando a biblioteca *ncurses*, por exemplo) ou gráfico (usando qualquer outra biblioteca, como *QT*, *GTK*);
- Criação de novas frentes. Na verdade este projeto já foi iniciado por Marcelo Rigon, que está desenvolvendo módulos de simulação de problemas ópticos. A idéia é que o software cresça e, cresça e, com isso, tenhamos um módulo mais geral de Física, por exemplo. Neste caso, os módulos Mecânica e Óptica seriam sub-módulos do Física.

6. Referências

1. J. B. Marion & S. T. Thornton, *Classical Dynamics of particles and systems*, 4a. Edição, Sauders College Publishing, Philadelphia, 1995.
2. K. R. Symon, *Mecânica*, 5a. Edição, Editora Campus, Rio de Janeiro, 1982.
3. H. M. Nussenzveig, *Física Básica*, 1a. Edição, Editora Edgar Blücher LTDA., São Paulo, 1981.
4. R. Johnsonbaugh & M. Kalin, *C for Scientists and Engineers*, Prentice Hall, New Jersey, 1997
5. T. Williams, & C. Kelley, *Gnuplot Reference Manual*, http://pi3.informatik.uni-mannheim.de/staff/mitarbeiter/westmann/gnuplot.html, 1993.
6. M. Abramawitz & I. A. Stegun, *Handbook of mathematical functions*, 1a. Edição, Dover Publications, New York, 1970

Avaliação como Forma de Incentivo ao Desenvolvimento de Software Livre em Ambiente Acadêmico

Francisco Reverbel

Departamento de Ciência da Computação
Instituto de Matemática e Estatística da Universidade de São Paulo

reverbel@ime.usp.br

Abstract. *This paper discusses criteria for evaluating the work of free software developers in an academic environment. It presents a set of parameters that are applicable to free software projects and can be used as indicators of significance. We suggest that traditional academic evaluation should be complemented by these parameters. This would be an effective way of fostering the innovative creation of free software in academia.*

Resumo. *Este artigo discute critérios de avaliação de atividades de desenvolvimento de software livre realizadas em ambiente acadêmico. É apresentado um conjunto de indicadores de relevância de projetos de software livre. Sugerimos que o sistema de avaliação acadêmica tradicional seja complementado com esses indicadores. Isso seria uma forma efetiva de incentivar a criação inovadora de software livre em ambiente acadêmico.*

1. Introdução

É crescente o reconhecimento da importância econômica e tecnológica do software livre. Formas de incentivo a projetos de pesquisa avançada que produzam software livre e/ou de fonte aberto estão sendo estudadas tanto no âmbito da Comissão Européia [European Working Group on Libre Software, 2000] como no do governo dos Estados Unidos [President's Information Technology Advisory Committee, 2000].

O software livre tem o potencial de contribuir para a redução do fosso tecnológico entre ricos e pobres [Kon, 2001]. Isto sugere que seu impacto positivo pode ser ainda mais forte sobre nações menos desenvolvidas. Para essas nações, o software livre não é apenas um modo de se evitar os custos do software fechado e a dependência de fornecedores monopolistas. Ele é, sobretudo, uma forma de acesso a tecnologias avançadas que dá a países menos desenvolvidos a possibilidade de participar efetivamente do processo de construção dessas tecnologias. É oportuno, portanto, que a questão dos incentivos ao software livre tenha começado a ser discutida em nosso país. Em particular, interessa-nos aqui a questão dos incentivos ao software livre no ambiente acadêmico [Simon, 2002].

O novo modelo de criação e distribuição de software coloca novos desafios para a Universidade brasileira: a difusão do software livre tanto no ambiente acadêmico como fora dele, a formação de recursos humanos capacitados para trabalhar com software livre e o desenvolvimento de software livre que contenha inovações tecnológicas. Este artigo aborda um tópico relacionado com o terceiro desses desafios: a avaliação como forma de incentivo ao desenvolvimento de software livre no ambiente acadêmico.

A criação de software tecnologicamente inovador é uma atividade de pesquisa que requer níveis muito altos de esforço e dedicação. Dependendo da natureza do software que se está desenvolvendo, os trabalhos de implementação podem ser inteiramente realizados por alunos de

pós-graduação ou de graduação, devidamente orientados por docentes universitários. Sistemas de software mais complexos, entretanto, requerem a intervenção direta de pessoas com maior conhecimento e experiência (os docentes) nos trabalhos de implementação. Surge aqui um problema: não há, no Brasil, nehum incentivo para que docentes universitários desempenhem tais tarefas, que tendem a consumir muito tempo. A avaliação docente é baseda na produção acadêmica tradicional, medida geralmente em termos de artigos publicados em revistas e em conferências científicas. O software inovador que um docente desenvolveu não é levado em conta por agências de fomento à pesquisa como CNPq e FAPESP, para efeito de concessão de verbas de pesquisa ao docente, nem tampouco por Universidades como a USP, para efeito da progressão do docente em sua carreira.

Nos países desenvolvidos é bem mais comum o envolvimento de docentes universitários em atividades de implementação de software livre.[1] Universidades americanas avaliam docentes das áreas de Ciência da Computação e Engenharia da Computação segundo os critérios explicitados em [Patterson et al., 1999], que levam em conta os "artefatos computacionais" (software, chips, etc.) produzidos pelo docente. Esse documento menciona alguns "indicadores de impacto" que podem ser usados na avaliação desse tipo de produção: número de *downloads* de um artefato de software, número de usuários, numero de visitas numa página Web, etc. No Brasil, entretanto, as agências de fomento à pesquisa e as Universidades relutam em atualizar seus critérios de avaliação. Duas dúvidas são recorrentes entre os avaliadores:

- Como reconhecer um projeto de software livre relevante e inovador?
- Como saber se uma pessoa efetivamente contribuiu para determinado projeto?

O autor se baseia em sua experiência de participação em dois projetos internacionais de software livre — o JBoss [Fleury and Reverbel, 2003, JBoss Group, 2003] e o JacORB [JacORB Team, 2003], ambos muito bem sucedidos — para oferecer respostas parciais a essas questões no restante deste artigo.

2. Indicadores de Relevância de Projetos de Software Livre

A relevância de um projeto de software livre é geralmente associada a um conjunto de indicadores. O vínculo entre a relevância do projeto e os indicadores relacionados a seguir é fortemente sugerido pela análise, *a posteriori*, dos projetos de software livre mais influentes e bem sucedidos, como Linux, Apache, GNU C/C++, GNU Emacs, Perl, PostgreSQL, FreeBSD, Samba, Sendmail e Bind.

Comunidade de participantes ativa e geograficamente dispersa. Ao longo de sua vida um projeto relevante tende a atrair um grupo de participantes ativos, com capacidade e disponibilidade para oferecer contribuições efetivas. O critério da dispersão geográfica reduz a probabilidade dos participantes terem sido motivados por fatores não necessariamente relacionados com a relevância do projeto. Ele elimina projetos cujos participantes são todos funcionários de uma mesma empresa (que pode ter interesse específico no projeto) ou alunos de um mesmo programa de pós-graduação (possivelmente orientados por um professor que participa do projeto), etc.

Dimensão do projeto. Projetos relevantes tendem a ter certo porte. O número de linhas de código fonte é em geral maior que cem mil e freqüentemente maior que um milhão.

Usuários. Todo projeto relevante tem usuários. A existência de usuários, além de gerar realimentação vital para o ciclo evolutivo do projeto, indica que ele atende a uma necessidade real e que ultrapassou o estado de protótipo. Tipicamente não existe uma contabilização precisa dos usuários de um programa livre. O volume de usuários é inferido a partir de outros

[1] A propósito: a versão final deste artigo foi gerada por um sistema de processamento de texto — o TEX — implementado pessoalmente pelo Prof. Donald Knuth, da Universidade de Stanford.

indicadores, como número de downloads e tráfego em listas de usuários. A publicação e comercialização de livros descrevendo o software desenvolvido (freqüentemente escritos por participantes do projeto) e a existência de estruturas comerciais de apoio aos usuários são também fatores relacionados com o número de usuários.

Tráfego em listas de discussão. Todo projeto relevante tem pelo menos uma lista de discussão na Internet. As mensagens enviadas para a lista são mantidas num repositório público e acessível a quem quer que seja. Muitos projetos tem mais de uma lista de discussão. Um arranjo típico utiliza uma lista para participantes do projeto, exclusivamente voltada para a discussão de aspectos internos do software coletivamente construído, e outra lista para os usuários desse software. Projetos relevantes têm um certo tráfego de mensagens em suas listas de discussão. Este tráfego varia de projeto para projeto, de dezenas de mensagens por mês até centenas de mensagens por dia.

Existência de literatura. A partir de um certo ponto de sua vida um projeto relevante se torna assunto de livros cujo mercado potencial é o conjunto de usuários do software produzido.

Comercialização de serviços de apoio. A partir de um certo ponto da vida de um projeto relevante surgem empresas que comercializam serviços de apoio aos usuários do software desenvolvido.

De acordo com esses indicadores, um projeto de software livre pode ser considerado relevante mesmo que não esteja alinhado com o estado da arte em sua área. (O Linux é freqüentemente enquadrado nessa categoria, por não empregar uma arquitetura tipo microkernel.) Como não contempla a questão da inovação tecnológica, esse conjunto de parâmetros é certamente incompleto para a avaliação da relevância de atividades de desenvolvimento de software livre desenvolvidas num ambiente acadêmico. Ele deve ser complementado por uma análise do grau de inovação e das contribuições para o estado da arte presentes no software desenvolvido, ou por parâmetros adicionais, especificamente voltados para uma avaliação acadêmica. Segundo [Patterson et al., 1999], a utilização de determinado software por outros pesquisadores, como instrumento ou objeto de pesquisa, é uma boa indicação da relevância acadêmica e da influência do projeto que gerou esse software.

3. Atribuição de Créditos em Projetos de Software Livre

O modo como os participantes de um projeto de software livre se organizam internamente varia bastante de projeto para projeto. Praticamente todos os projetos bem sucedidos adotaram como organização social alguma forma de meritocracia. O mérito de cada participante é baseado no valor de suas contribuições para o projeto. O funcionamento da organização repousa portanto sobre o reconhecimento coletivo (que necessariamente inclui uma avaliação) da contribuição de cada um.

A correta atribuição de créditos pelas contribuições dos participantes é uma condição imprescindível para a harmonia interna da estrutura social do projeto. É extremamente difícil uma apropriação indevida do crédito pelo trabalho realizado por outrem. Na improvável hipótese de que tal apropriação ocorra, é virtualmente impossível que ela passe despercebida. Além disso, num projeto com certa importância, mesmo a cessão voluntária de crédito (prática muito difícil de se eliminar num ambiente acadêmico tradicional) é inviabilizada pelo fato dos trabalhos serem realizados em público. A história completa das modificações nos arquivos fonte é mantida num repositório acessível a todos os participantes (e a toda e qualquer pessoa, em muitos projetos). Esse repositório é utilizado diariamente pelos membros ativos do projeto. Em muitos projetos o repositório de arquivos fonte é configurado para enviar automaticamente uma mensagem informativa a todos os participantes sempre que algum arquivo for modificado. Tais mensagens contém os nomes dos arquivos modificados, as alterações efetuadas e o participante responsável pelas alterações.

A quase totalidade dos projetos tem pelo menos duas formas básicas de reconhecimento pelas contribuições de seus participantes:

Direito de escrita no repositório de arquivos fonte. É a forma mais elementar de reconhecimento. Um "*committer*" é um participante que tem o direito de efetivar alterações ("*commit*") no repositório de fontes do projeto. Novos participantes somente se tornam *committers* depois de demonstrarem um certo nível (que varia bastante entre projetos) de dedicação e de competência. Do ponto de vista de segurança, é indesejável a existência de *committers* que não estejam efetivamente contribuindo para o projeto. Por esse motivo, muitos projetos adotam alguma política de revogação do direito de escrita de participantes inativos.

Divulgação do nome do participante. Os nomes dos participantes são geralmente publicados na página do projeto, com grau de destaque proporcional à importância de suas contribuições.

Além dessas formas elementares de reconhecimento, projetos com uma estrutura organizacional mais complexa adotam esquemas mais elaborados para definir algum tipo de diferenciação hierárquica entre seus participantes.

4. Conclusão

A criação inovadora de software livre em ambiente acadêmico pode ser incentivada significativamente com uma acão das agências de fomento à pesquisa e das Universidades: a atualização dos critérios de avaliação de pesquisadores e docentes, de modo a levar em conta o software livre por eles produzido. Este artigo é uma contribuição nesse sentido. Não temos, no entanto, a pretensão de ter identificado um conjunto de parâmetros adequado para avaliar todo e qualquer projeto de software livre. Por estar fortemente relacionada com a existência de usuários, boa parte dos parâmetros identificados neste artigo não capta a relevância de projetos que ainda se encontrem nos estágios iniciais de seu ciclo evolutivo. Mesmo assim, acreditamos que um conjunto de indicadores nos moldes do aqui proposto pode enriquecer muito o processo de avaliação da produção acadêmica.

Referências

European Working Group on Libre Software (2000). "Free Software / Open Source: Information Society Opportunities for Europe?". European Commission Information Society. `http://eu.conecta.it/paper.pdf`.

Fleury, M. and Reverbel, F. (2003). The JBoss Extensible Server. In *Middleware 2003 — ACM/IFIP/USENIX International Middleware Conference*, volume 2672 of *LNCS*, pages 345–374. Springer-Verlag. `http://www.ime.usp.br/~reverbel/papers/jboss-mw2003.pdf`.

JacORB Team (2003). JacORB web site. `http://www.jacorb.org`.

JBoss Group (2003). JBoss web site. `http://www.jboss.org`.

Kon, F. (2001). O Software Aberto e a Questão Social. Relatório Técnico RT-MAC-2001-07, Departamento de Ciência da Computação, IME-USP. `http://www.ime.usp.br/~kon/papers/RT-SoftwareAberto.pdf`.

Patterson, D., Snyder, L., and Ullman, J. (1999). "Evaluating Computer Scientists and Engineers for Promotion and Tenure". Best Practices Memo, Computer Research Association. `http://www.cra.org/reports/tenure_review.html`.

President's Information Technology Advisory Committee (2000). "Developing Open Source Software to Advance High End Computing". Report to the President, Executive Office of the President of the United States. `http://www.itrd.gov/pubs/pitac/pres-oss-11sep00.pdf`.

Simon, I. (2002). "O Software Livre precisa de incentivos? Aonde? De que tipo? Como proceder?". Palestra proferida no I Forum Nacional de Software Livre em Instituições de Ensino Superior (SOLIES), Universidade Federal de São Carlos. `http://www.ime.usp.br/~is/aula/incentivos/`.

ENSINET: Uma solução Software Livre para Apoio ao Ensino Semi-Presencial utilizando a Internet*

Eduardo Nunes Ferreira Bastos[1], Antônio Carlos da Rocha Costa[1], Graçaliz Pereira Dimuro[1]

[1]Escola de Informática
Universidade Católica de Pelotas (ESIN/UCPel)
Rua Felix da Cunha 412, 96010-000, Pelotas, RS

{eduardob, rocha, liz}@atlas.ucpel.tche.br

Abstract. *This article presents the current stage in the development of the ENSINET, an environment for web-based education, which is being built at UCPel. The present work describes the adopted basic architecture and shows future perspectives for the system.*

Resumo. *Este artigo apresenta a etapa atual do desenvolvimento do sistema ENSINET, um ambiente de apoio ao ensino semi-presencial utilizando a Internet, em construção na UCPel. O presente trabalho descreve a arquitetura básica adotada e mostra perspectivas futuras para o sistema.*

1. Introdução

O Projeto ENSINET/UCPel [Dimuro et al. 1998, Dimuro et al. 2000] está desenvolvendo o ambiente ENSINET segundo a concepção de software livre. Este ambiente tem como finalidade automatizar a disponibilidade de materiais de apoio, condução de discussões, apresentação de conteúdos ou aplicações em forma de páginas na web, entre outras diversas ferramentas e recursos, para suporte ao ensino semi-presencial. O site do projeto e do sistema podem ser acessados, respectivamente, em http://gmc.ucpel.tche.br/ensinet e http://ensinet.ucpel.tche.br.

Atualmente, está em desenvolvimento a segunda versão deste sistema. Foram adaptadas algumas funcionalidades oriundas da primeira versão, e construídas novas ferramentas e módulos, numa abordagem 100% orientada a objetos.

O sistema ENSINET foi implementado no ZOPE - Z Object Publishing Environment [ZOPE-Corporation 2003, Latteier et al. 2001], um servidor de aplicações web multiplataforma integrado a um banco de dados transacional orientado a objetos, e para construção dos módulos, foi utilizada a linguagem de programação Python [Python Software Foundation 2003], a qual encapsula na sua estrutura de classes, o funcionamento interno da maioria das funcionalidades avançadas necessárias a construção de um website, como: renderização de documentos dinâmicos, reutilização e transparência de código, aspectos de segurança, entre outras.

Uma descrição em detalhes do sistema ENSINET só pode ser dada num relatório extenso, incluindo ilustração de códigos, diagramas e modelos. O objetivo deste artigo é descrever de forma sucinta, a implementação das camadas inferiores e introduzir conceitos para perpectivas futuras do sistema. A seção 2. apresenta uma visão

* Este trabalho tem financiamento do CNPq e da FAPERGS.

geral do sistema. A seção 3. descreve objetos temporais versionados e monitorados, e de acesso inseridos na camada denominada Z-TVM. A seção 4. apresenta conceitos sobre a camada Z-ServAprend. As conclusões e considerações finais estão na seção 5.

2. Descrição do Sistema ENSINET

O sistema ENSINET[1] está sendo desenvolvido com base em uma arquitetura robusta que permite aos desenvolvedores implementarem módulos independentes, de forma rápida e fácil. Estes módulos, denominados produtos, são programados na linguagem Python e inseridos no contexto do framework ZOPE, para publicação na forma de conteúdo HTML na web.

A criação desses módulos, possibilitou a implementação do sistema em camadas, tornando o trabalho de criação e atualização muito mais ágil e eficaz. Foram modeladas várias camadas, desde a de mais baixo nível, provida por objetos do próprio ZOPE, até camadas como: API de Serviços de Ensino-Aprendizagem e Serviços de Sistema.

3. Camada Z-TVM

Esta camada, denominada Z Temporal Version Model, contém tanto os objetos fundamentais concebidos pelo ZOPE, quanto os objetos temporais versionados e monitorados e de acesso, inseridos no contexto do Banco de Objetos TVM.

Os objetos básicos fornecidos pelo framework ZOPE, permitem executar tarefas distintas, e assumem um desses três papéis:

1. **Conteúdo**: objetos do ZOPE são como documentos, imagens e arquivos com tipos diferentes de dados textuais e binários.
2. **Apresentação**: são fornecidas duas linguagens de programação, para facilitar o gerenciamento de conteúdo: DTML – Document Template Markup Language e ZPT – Zope Page Templates.
3. **Lógica**: disponibiliza algumas facilidades, como: DTML, Python e Pearl. A idéia é separar a lógica de programação da interface que será mostrada para o usuário final.

Para adicionar aspectos de tempo e versão nos objetos básicos providos pelo ZOPE, foi proposta uma modificação ao Modelo Temporal de Versões [Moro 2001], ilustrada na Fig. 1, criando-se conceitos de objeto temporal não-versionado, temporal versionado e não temporal versionado. Este modelo baseia-se nos conceitos de versão e tempo para armazenar as versões do objeto, os seus tempos de vida, e o histórico das alterações feitas nos valores dos atributos e relacionamentos dinâmicos. Dessa maneira, o tempo é associado ao objeto, à versão, aos atributos e aos relacionamentos.

A proposta do Modelo de Objetos Temporais Versionados [Nieves 2003] ainda não foi implementada para uso no ambiente, mas é de extrema importância para ferramentas futuras de monitoração e análise, visto que será possível, por exemplo, gerar estatísticas de acesso a uma determinada disciplina ou avaliar de forma mais criteriosa o uso do sistema pelos alunos. Foram realizados estudos preliminares e conceituações sobre uma monitoração de usuários informada [Miranda 2003], para implantação no sistema ENSINET. Essa monitoração permitirá, por exemplo, a inclusão

1 A versão inicial do sistema ENSINET está descrita em [Miranda et al. 2001], incluindo exemplos de casos de uso.

de sites internos adaptativos, como proposto em [Dimuro et al. 2002].

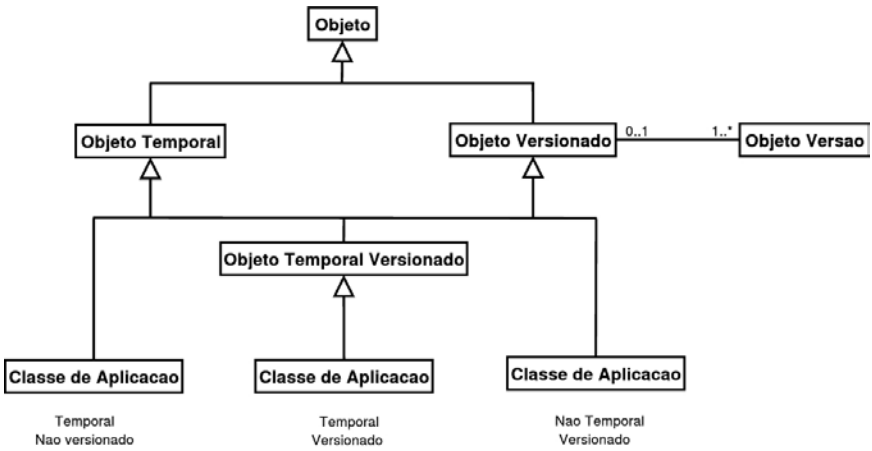


Figura 1. Proposta do Modelo de Objetos Temporais Versionados

4. Camada Z-ServAprend

Esta camada, denominada Z Serviços de Aprendizagem, contém três componentes que se correlacionam para prover ferramentas e recursos disponibilizados no ambiente, bem como: banco de usuários, banco de componentes de aprendizagem e uma API de Serviços de Aprendizagem.

4.1. Banco de Usuários

A camada do Banco de Usuários do sistema ENSINET é formada pela infraestrutura do banco, onde são especificados e programados vários métodos que fazem a conexão com o banco de dados ZODB, e por instâncias de usuário. Foi imprescindível a implementação deste, para adicionar novos conceitos aos usuários básicos que o ZOPE provê, como por exemplo, Perfil Acadêmico.

O usuário ENSINET possui métodos e atributos próprios, como por exemplo: contexto atual no sistema, histórico de uso do sistema e perfil acadêmico, sendo especializado em 6 classes: administrador, aluno, professor, monitor, grupo e visitante.

O administrador tem controle total do sistema, ficando a cargo dele gerenciar o diretório de usuários, disciplinas, entre outras ferramentas e recursos disponíveis. O professor gerencia os conteúdos apresentados aos alunos e monitores. O aluno pode visualizar o conteúdo exposto pelo professor e utilizar recursos disponibilizados na disciplina. Monitor, é um aluno com mais privilégios, estando apto a gerenciar alguns módulos da disciplina, liberados pelo professor. Grupo é um conceito de composição de usuários alunos, professores, etc., que participam de uma determinada tarefa. Por fim, o visitante é um usuário não cadastrado que pode apenas acessar e visualizar conteúdos previamente estabelecidos pelos administradores e professores das disciplinas.

4.2. Banco de Componentes de Aprendizagem

Este banco de componentes de aprendizagem é constituído por uma infraestrutura e vários componentes de aprendizagem. Este componente de aprendizagem representa os recursos disponibilizados no ambiente, como: material de apoio, atividades, mural de avisos, etc., sendo formado por objeto de metadados (descritor) e objeto de aprendizagem (dado propriamente dito).

5. Considerações Finais

A utilização dos recursos da Internet associada às vantagens oferecidas por softwares livres, como ZOPE e Python, contribuem de forma satisfatória para o ensino integrado, destacando-se a experiência de ensino semi-presencial. Espera-se que, com a utilização de software livre, o desenvolvimento e utilização do sistema possa ser compartilhado com a comunidade ZOPE, visando a ampliação de suas funcionalidades e de sua aplicação como ferramenta de apoio ao ensino semi-presencial.

As próximas etapas do trabalho envolverão o desenvolvimento de uma monitoração e avaliação do ambiente, a ser realizada de forma continuada e interativa. Isso possibilitará a construção de ferramentas importantes para auxiliar no processo de avaliação das interações realizadas entre os usuários e os objetos do sistema.

6. Referências

Dimuro, G. P., Costa, A. C. R. and Rodrigues, F. P. M. (1998) “Projeto ENSINET”, ESIN/UCPel, http://gmc.ucpel.tche.br/ensinet, May.

Dimuro, G. P., Costa, A. C. R. and Rodrigues, F. P. M. (2000) “Uma experiência de ensino integrado dos fundamentos matemáticos da ciência da computação”, Revista Brasileira de Informática na Educação, n. 7, p. 65-75.

Dimuro, G. P., Palazzo, L. A. M., Costa, A. C. R. and Miranda, R. M. (2002) “Sites Internos Adaptativos para o Sistema Ensinet”, In: III Workshop de Software Livre, Porto Alegre, SBC, p. 102-105.

Latteier, A. and Pelletier, M. “The Zope Book”, New Riders Publishing, 2001.

Miranda, R. M., Dimuro, G. P. and Costa, A. C. R. (2001) “Um Ambiente de Suporte ao Ensino Integrado dos Fundamentos Matemáticos da Ciência da Computação utilizando o ZOPE”, In: II Workshop de Software Livre, P. Alegre, SBC, p. 40-42.

Miranda, R. M. “Monitoração de Usuários Informada: Conceituação e Estudo Preliminar para sua Implantação no Ambiente ENSINET”, Porto Alegre, PPGC/UFRGS, 2003.

Moro, M. M. “Estudo da união de um modelo orientado a objetos com versões e modelos temporais”, T.I n. 953, Porto Alegre, PPGC/UFRGS, 2001.

Moro, M. M. “Modelo Temporal de Versões”, Porto Alegre, PPGC/UFRGS, 2001.

Nieves, I. V. and Costa, A. C. R. (2003) “Estudo e Adaptação de um Modelo de Objetos Temporais Versionados para o Sistema ENSINET”, ESIN/UCPel, http://gmc.ucpel.tche.br/ensinet, May.

Python Software Foundation (2003), http://www.python.org, May.

ZOPE-Corporation (2003) ZOPE Community, http://www.zope.org, May.

CRIANET:
uma plataforma de software para crianças na Internet

Sílvia MEIRELLES LEITE[1], Patrícia BEHAR[1], Tiago CUNHA[1], Maicon BRAUWERS[1]

[1]Núcleo de Tecnologia Digital aplicada à Educação - Faculdade de Educação
Pós-Graduação em Informática na Educação - Pós-Graduação em Educação
Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Paulo Gama s/n - Prédio 12201 - Centro - 90046-900 - Porto Alegre, RS, Brasil

silvia@vetorial.net, pbehar@terra.com.br, tiago_cunha@hotmail.com, maicon@edu.ufrgs.br

Resumo. *O CRIANET (CRIAnça na interNET) é um protótipo de plataforma de software voltado ao trabalho coletivo com crianças na Internet. Este integra ferramentas para a comunicação síncrona e assíncrona e oportuniza o encontro dos participantes através da rede. O objetivo deste projeto é o desenvolvimento de um software livre para educação com crianças. Foi selecionado o sistema operacional Linux, como servidor e plataforma de desenvolvimento e, para a construção do sistema, utilizou-se a linguagem de programação PHP. O CRIANET é aberto para a criação de versões próprias.*

Abstract. *The CRIANET (CRIAnça na interNET - Child in The Internet) is a prototype of software platform focalized in the collective work with students of initial series. This integrates tools for the synchronous and asynchronous communication and gives opportunity to the participants' encounter through the net. The main goal of this project is the development of a free software to education with children. It was select the Linux operation system as the host and development platform and, to construct the system, it was use the PHP program language. The CRIANET is open for creation of self-version.*

1. Introdução

Este trabalho tem como proposta apresentar o protótipo do CRIANET (CRIAnça na interNET), que está disponível em http://www.nuted.edu.ufrgs.br/~crianet. Esta é uma plataforma de software livre desenvolvida com a finalidade de oportunizar um espaço na Internet para o trabalho em grupo com crianças. Dentro dessa perspectiva, propõe-se a constituição de um ambiente virtual através do CRIANET, de forma que seus usuários possam participar ativamente das trocas que o compõem.

A estrutura do CRIANET integra ferramentas para comunicação síncrona e assíncrona e registra as contribuições postadas pelos participantes, possibilitando um diálogo permanente entre os sujeitos. Este protótipo foi implementado por uma equipe interdisciplinar composta por profissionais da informática, educação e comunicação, o que foi feito pelo Núcleo de Tecnologia Digital aplicada à Educação, da Faculdade de Educação da Universidade Federal do Rio Grande do Sul (NUTED/FACED/UFRGS), a partir do framework ROODA_DEVEL[1]. Esse protótipo faz parte do projeto ROODA[2] (Rede cOOperativa De

[1] O ROODA_DEVEL (Behar et ali, 2002) é um *framework* para o desenvolvimento de ambientes virtuais para a Educação à Distância (EAD).

[2] O Projeto ROODA recebeu um prêmio, no ano de 2002, do Programa de Apoio à Pesquisa em Educação a Distância (PAPED), vinculado a CAPES/SEED/ MEC, sendo o primeiro colocado entre os cinco projetos selecionados em todo o país. A premiação foi destinada a materiais didáticos multimídia para uso em cursos ou disciplinas, ministrados presencialmente ou a distância, via Internet.

Aprendizagem)[3], que tem como pressuposto a filosofia do software livre, portanto, seu código fonte é aberto (Behar et alli, 2001). A construção da plataforma envolveu tanto sua programação quanto sua interface.

Para o desenvolvimento do CRIANET, foi feito um estudo exploratório com crianças, alunos das séries iniciais do Ensino Fundamental, sobre a acessibilidade e navegabilidade de sites disponíveis na Internet (Meirelles Leite; Moresco; Behar, 2002). Os dados coletados nesse primeiro momento serviram como norteadores para o sua elaboração e implementação, tendo-os como referência para ampliar as possibilidades de encontro entre os seus usuários.

Também foram determinantes para as escolhas feitas no decorrer de sua construção os pressupostos teóricos construtivistas adotados, sendo a principal referência Piaget (1973). Com isso, busca-se ter na plataforma de software um meio para aproximar as crianças na Internet, favorecendo a coletividade e privilegiando as trocas dentro do contexto escolar.

Assim, debater o desenvolvimento de uma plataforma de software livre é buscar tanto nos pressupostos teóricos e metodológicos que se tem como base, quanto nas questões técnicas, a fundamentação para elaborá-la. Esse princípio prevaleceu na construção do CRIANET, o que foi ressaltado por se tratar de um produto voltado à educação.

2. Delineando o CRIANET

O CRIANET foi construído a partir da idéia de metáfora[4]. A escolha de temática usada baseou-se nos relatos das crianças que foram coletados no estudo exploratório. Ao propor a ambientalização deste espaço em uma casa valorizou-se o aspecto coletivo, de forma que os próprios participantes habitem-na e se tornem responsáveis por sua manutenção. Ressalta-se que o CRIANET continua em desenvolvimento, portanto nem todas as ferramentas idealizadas foram implementadas.

Quando o sujeito entra no CRIANET, ele acessa diretamente o grupo no qual está cadastrado, tendo na casa um espaço que é compartilhado por todos os "moradores". Sua área de trabalho (Figura 1) é caracterizada como uma casa e seu entorno. Com isso, pode-se contorná-la ou entrar nela, encontrando seus cômodos. Esses podem ser acessados entrando pela porta da casa e navegando dentro dela, através das janelas ou do dropdown. As ferramentas podem ser acessadas por um menu superior, nos cômodos ou no dropdown.

Essas opções variadas de acesso tiveram como princípio proporcionar aos sujeitos construir seus caminhos, valorizando diferentes processos de apropriação. Nisso, utilizou-se tanto uma lógica cristalizada na Internet (navegação pelo menu superior e dropdown), quanto uma lógica calcada no intuitivo. Quando é empregada essa expressão, tem-se como base o pensamento intuitivo apresentado por Piaget (1973). Este é caracterizado por valorizar as antecipações e reconstituições representativas.

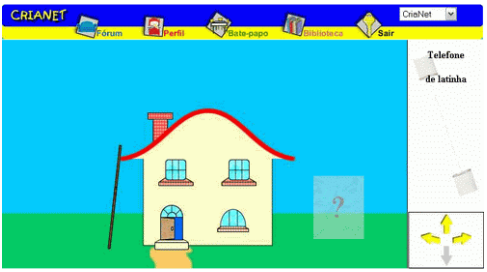


Figura 1: Área de Trabalho do CRIANET, primeira tela acessada após o login.

[3] Ambiente virtual desenvolvido pelo NUTED/FACED/UFRGS (Núcleo de Tecnologia Digital aplicada à Educação), coordenado pela Profª Drª Patrícia Alejandra Behar. Disponível em: http://rooda.edu.ufrgs.br/

[4] A idéia de metáfora tem como base Valentini e Fagundes (2001), buscando convidar os participantes a compartilhar o espaço disponível.

A disposição dos links para as ferramentas tem o objetivo de ampliar as possíveis interações entre os participantes. Como foi observado no estudo exploratório que algumas crianças inicialmente procuravam clicar no centro da tela e, principalmente, nas imagens com movimento, optou-se por colocar animação onde existissem atalhos. Da mesma forma, escolheu-se privilegiar espaços com desenhos em conjunto com a escrita, a fim de que as crianças possam ler tanto as imagens, quanto as palavras, sem restringi-las a textos extensos. Um exemplo disto está nas portas. Cada uma delas apresenta uma placa indicando a qual cômodo levará e a cor do mesmo; ao passar o mouse sobre ela, movimenta-se, ficando entreaberta, e no canto aparece a mesma cor das letras (Figura 2).

As imagens da casa são em duas dimensões, mas sua navegação simula uma terceira dimensão, possibilitando vê-la de mais de um ponto de vista. Dentro disto, explorou-se a idéia de proporção dos espaços e dos móveis, incluindo a profundidade e a perspectiva.

Os cômodos encontrados são: sala, cozinha, quarto, biblioteca, corredor e um outro cômodo com um ponto de interrogação para ser construído pelos moradores. Também se pode acessar os lados da casa, os fundos e o telhado. Esse deslocamento pode ser feito através das setas que apontam as direções ou atravessando as portas, janelas e passagens secretas. Em cada cômodo é encontrada uma ferramenta, sendo que ambos são caracterizados pela mesma cor.

Figura 2: Sala do CRIANET, com acesso ao fórum e aos outros cômodos da casa.

As ferramentas disponíveis no presente protótipo são:

1- Fórum: encontrado na sala, foi adaptado para o debate síncrono e assíncrono. Para tanto, foi utilizado como referência o princípio apresentado pelo *for-chat* (Axt et alli 2002)[5], mas não é igual a ele. As mensagens são colocadas uma embaixo da outra, seguindo uma ordem cronológica de envio, não são separadas por temas ou tópicos e ficam todas na mesma tela.

2- Perfil: encontrado no quarto, é onde o morador pode se apresentar e colocar sua imagem. Cada participante só pode modificar o seu perfil, porém, tem acesso ao perfil dos seus colegas. A cor escolhida no perfil, se torna a cor do nome no fórum.

3- Biblioteca: encontrada no cômodo que também se chama biblioteca, é usada para a publicação dos arquivos do grupo. Nela, é possível enviar arquivos ou visualizar os que foram enviados anteriormente. Os mesmos podem ser copiados, mas não podem ser apagados.

4 – Banco de Figuras: fora e dentro da casa, nos locais onde são encontrados pontos de interrogação. Nesses espaços, o participante pode inserir uma figura, enviada por ele ou por algum colega. Nessa ferramenta, os participantes montam seu banco de figuras, sendo este compartilhado por todos do grupo. Quando é colocada uma nova figura no lugar do ponto de interrogação, ela fica visível para todos os participantes, de forma que eles participem da constituição do ambiente coletivamente. Essa ferramenta também é usada no cômodo a ser construído pelos membros do grupo.

Nesta perspectiva, busca-se um espaço que privilegie as contribuições das crianças, rompendo com uma estrutura unilateral. As contribuições enviadas são tidas como referência para a constituição da

[5] Ferramenta desenvolvida pelo LELIC/FACED/UFRGS, descrita como "uma composição híbrida a meio caminho entre um fórum, um chat e um mural eletrônico (um *for-chat*), permitindo o registro das contribuições, tanto síncronas, quanto assíncronas" (Axt et alli, 2002, p.3).

coletividade no CRIANET, sem que prevaleça somente a fala do professor e sem a preocupação de editar as contribuições dos alunos. Com isso, é enfatizado o processo.

Dentro dos aspectos técnicos do CRIANET, tem-se como princípio a filosofia do software livre, estando aberto para a contribuição da comunidade. Dessa forma, foi selecionado o servidor de http Apache, a linguagem PHP e o banco de dados MySQL, rodando numa máquina com o sistema operacional Linux. Como sua programação foi feita a partir do ROODA_DEVEL, ele utiliza as camadas de abstração desse *framework*, que é dividido nos conjuntos de classes de sistema, interface e ferramentas (Behar et al, 2002). O primeiro representa entidades como usuários e grupos de trabalho, cada qual com suas regras de acesso ao banco de dados. O segundo, a capacidade de utilizar templates e widgets para exibir páginas. Enquanto o terceiro, ferramentas como fórum, upload e controle de logon. Estas classes visam tornar mais fácil e rápido desenvolver a plataforma, ao mesmo tempo que aumentam a segurança e portabilidade do código.

3. Considerações Finais

Ao ter como base uma plataforma de software que integre ferramentas para a comunicação através da Internet e o favorecimento da constituição da coletividade com crianças, busca-se ampliar as possibilidades de encontro na Rede. Dentro disso, o CRIANET foi planejado tendo como proposta facilitar o acesso às contribuições dos colegas. Isso envolve sua estrutura não-linear na disponibilização das ferramentas e o uso de recursos figurativos na efetivação das trocas entre os participantes.

O processo de construção dessa plataforma é contínuo, o que envolve pequenos ajustes em todos os recursos até então disponibilizados, para um melhor desempenho, e a implementação de novas ferramentas. Atualmente estão sendo desenvolvidas duas novas ferramentas, usadas para a comunicação síncrona. Uma delas é o bate-papo, que será implementado na cozinha. A outra é o telefone de latinha[6], que lista os usuários conectados ao mesmo tempo (de todos os grupos cadastrados) e possibilita a comunicação entre eles, dentro do princípio da relação um-um.

Em razão de o código fonte ser aberto, é possível que outros interessados colaborem na construção de uma plataforma mais eficiente, de forma que eles próprios possam realizar alterações. Com isso, tem-se a vantagem do CRIANET não ficar restrito a realidades isoladas, podendo ser adequado conforma as necessidades de outros usuários. Essa possibilidade é enfatizada por ele ser voltado ao uso educacional e por buscar-se uma integração com o contexto escolar.

4. Referências Bibliográficas

AXT, Margarete; et alli. (2002) **Produção coletiva em rede: é possível avaliar?**. (mimeo).

BEHAR, Patrícia; et al. (2002) **ROODA DEVEL: Uma Proposta de Framework para a Construção de Plataformas de Educação à Distância**. In: III Workshop de Software Livre WSL2002.

_____. (2001) ROODA – Rede cOOperativa De Aprendizagem – Uma plataforma de suporte para aprendizagem à distância. **Revista Informática na Educação: Teoria & Prática**, Porto Alegre, v.3, n. 2, p. 87-96.

MEIRELLES LEITE, Sílvia; MORESCO, Sílvia; BEHAR, Patricia. (2002) **A Interação de Crianças e Adolescentes em Ambientes Virtuais: identificando fatores de acessibilidade e navegabilidade.** In: XIII Simpósio Brasileiro de Informática na Educação - XIII SBIE. São Leopoldo, R.S. p. 658.

PIAGET, Jean. (1973) **Estudos Sociológicos**. Rio de Janeiro: Forense.

VALENTINI, Carla Beatris; FAGUNDES, Lea. (2001) Ambientes Virtuais de Aprendizagem: sistema, organização e interação. **Revista Informática na Educação: Teoria & Prática**, Porto Alegre, v.4, n.2, p.109-117.

[6] Essa ferramenta foi desenvolvida a partir do Finder, disponibilizado no ROODA (Behar et alli 2001).

ETC - Editor de Texto Coletivo
Um software livre para auxiliar a escrita coletiva através da Web

Profª . Dra. Patricia Alejandra Behar
pbehar@terra.com.br

Maicon Brauwers
Juliano Bittencourt
Silvia de Oliveira Kist
Aluísio Pinheiro
Daisy Schneider

Maria Carolina Colombo
Silvia Ferreto Moresco
Maira Bernardi
Alexandra Lorandi Macedo

Núcleo de Tecnologia Digital aplicada à Educação - NUTED
Faculdade de Educação - FACED
Pós-Graduação em Informática na Educação (PGIE) e em Educação (PPGEdu)
Paulo Gama s/n – Predio 12201 – Sala 1002 – 90046.900 – Centro - RS
Universidade Federal do Rio Grande do Sul – UFRGS

RESUMO

A idéia do ETC surgiu a partir da necessidade de criar e construir textos coletivos com alunos de graduação e de pós-graduação da Faculdade de Educação da UFRGS. A opção para o seu desenvolvimento foi baseada em LINUX, refletindo o interesse do grupo em utilizar uma plataforma livre como produtos de software livre. Sua implementação foi apoiada no ROODA_DEVEL, que é um Framework para Construção de Plataformas de Educação à Distância. O presente estudo apresenta, brevemente, os princípios técnicos e educacionais que fundamentam o ETC (versão atual 1.0), suas funcionalidades, as experiências até então realizadas que perpassaram "as paredes" da Universidade e se estenderam em escolas de ensino fundamental e médio. Finalizando, apresentam-se os trabalhos atuais e futuros acerca do seu desenvolvimento e aplicação.

ABSTRACT

The conception of ETC comes from the necessity for writing colective texts with graduate and post-graduate students of Education College from UFRGS. Because of the group interest in using freee software products Linux was chosen for the ETC development. The implementation of ETC was based on ROODA-DEVEL, a framework for distance learning plataform construction.The present study presents, briefly, technicals and educational principles that support the ETC (up-to-date version 1.0), its operational functions, the experiences done so far and which went through the "walls" of the University and reached elementary and high schools. Finally, are presented the present and future works about its development and use.

1. Introdução

O ETC começou a ser desenvolvido no início de 2002 pelo NUTED (Núcleo de Tecnologia Digital aplicada à Educação), na Faculdade de Educação, com o objetivo de propiciar a escrita coletiva/cooperativa através da rede. A necessidade deste ambiente surgiu a partir das dificuldades observadas na elaboração de trabalhos coletivos, mais precisamente nos cursos a distância, por meio da Web.

Até o momento, para esse tipo de trabalho, estava se utilizando o Equitext (Rizzo, 2000), um editor de texto coletivo, construído por uma equipe do Programa de Pós Graduação em Informática e Educação (PGIE/UFRGS). Mas, por se tratar de um software desenvolvido na linguagem *Perl*, não houve possibilidade de adaptá-lo as vicissitudes dos trabalhos do NUTED, que trabalha fundamentalmente com *GNU/LINUX* e linguagem *PHP*. Portanto, optou-se pela construção de uma nova ferramenta construída sobre o framework ROODA DEVEL (Behar, 2002a), possibilitando a sua integração com todos os software construídos sobre este framework, de forma a facilitar o trabalho do grupo.

Cabe enfatizar que, dentro dos recursos do ETC, que se distinguem daqueles encontrados no Equitext, estão a inserção de imagens além de textos, a não existência de hierarquia entre os usuários dentro de um texto e a possibilidade de construção de comunidades dentro do ambiente. Além disso, o ETC foi desenvolvido dentro da filosofia do Software Livre, propiciando que qualquer pessoa possa utilizá-lo, bem como, contribuir com o código fonte ou adaptá-lo às suas necessidades.

Trata-se de um editor de texto coletivo e, o chamado *coletivo* nesta abordagem, depende do tipo de interação que é mantida, neste caso, trata-se da interindividual que ocorre entre os participantes de uma atividade, ou seja, da elaboração conjunta de um texto, de forma síncrona e/ou assíncrona. Logo, a fundamentação teórica do presente artigo foi baseada na teoria piagetiana sobre a definição dos tipos de interação ocorridas entre os sujeitos, bem como os conceitos envolvidos em um texto coletivo, segundo Morin (2000) e Bakthin (2000).

O presente artigo traz uma abordagem sobre os pressupostos técnicos, conceitos pedagógicos, funcionalidades e relatos de experiências aplicadas até o momento. O artigo que contém os pressupostos teóricos e os dados coletados nas diversas experiências encontra-se detalhado em (Behar, 2003).

2. Princípios técnicos e pedagógicos que fundamentam o ETC

O ETC foi programado utilizando o framework ROODA DEVEL (Behar, 2002a), que é um conjunto de classes em PHP, desenvolvidas para atender as necessidades dos ambientes virtuais de aprendizagem, acoplado a um wrapper objeto-relacional. Concretamente, o framework propicia funções comuns a maioria dos ambientes virtuais de aprendizagem, como, por exemplo, autenticação de usuários, gerenciamento de grupos, diferenciação dos papéis dentro de um grupo, ferramentas básicas de comunicação (fórum, chat, correio eletrônico) e um sistema de gerenciamento de interfaces.

Ferramentas como o ETC, que são disponibilizadas em ambientes virtuais de aprendizagem proporcionando uma construção coletiva, exigem dos interagentes uma relação lingüística aberta, pois não existe outra possibilidade senão a escrita para tal construção. Através das interposições do texto coletivo, pode-se revelar a trama e a identidade do grupo, partindo da tradução feita das mais variadas contribuições sobre o mesmo tema (Morin, 2000). Esta trama de contribuições, permite o acontecimento de uma autoregulação, de uma reconstrução, até mesmo uma reavaliação contínua do processo de escrita de cada um dos sujeitos envolvidos. O texto é perfeitamente mutável e passível de transformações, sendo direcionado para as vivências e conhecimentos prévios do tema por cada um, a qualquer momento. O que deve ser considerado verdadeiramente relevante pelos sujeitos interagentes, é o objetivo centrado na construção coletiva, na troca de idéias e nas proposições sobre as mais diversas vivências. Pode-se assim dizer, que os textos coletivos permitem a superação do individual enquanto "eu" solitário, abrindo-se para o "eu" solidário. No âmbito de conjunto, pode gerar também uma desorganização temporária de idéias e superação na organização de diferentes pontos de vista.

Além de permitir a cooperação, a troca síncrona ou assíncrona pelas mais diversas linhas de pensamento, o ETC permite a metacognição, para que o sujeito possa retornar seu pensamento e, se necessário, refazê-lo sob novas perspectivas. O pensar sobre o pensar, traz novas possibilidades de construção e de pensamento. Logo, a produção coletiva viabilizada através desta ferramenta estabelece o que Bakhtin (2000) denomina de polifonia, ou seja, a produção do texto coletivo como um entrelaçamento de vozes, que pode se constituir dentro de uma nova lógica de tempo e de espaço. O espaço criado pelo ETC para construção de textos coletivos implica, portanto, em condições de ação e reflexão entre os sujeitos, originando a transformação de cada um e destes em relação ao grupo. As múltiplas trocas favorecem a escrita dinâmica e não pré-idealizada pelos interagentes, por isso prevêem uma diferenciada organização da estrutura coletiva. Esta organização não significa impor ordens de contribuição, como um depois do outro, mas uma organização aberta, que se adapte aos participantes, possibilitando formas pertinentes de se fazer uso das contribuições e também favorecendo o reconstruir de tantas outras, em busca de um todo coerente e significativo.

Portanto, o ETC, vem proporcionar uma escrita coletiva, dando ênfase à interação e cooperação (Piaget, 1973). Com a preocupação centrada no acompanhamento do processo de pensamento de cada sujeito integrante da construção de textos coletivos, o ETC traz consigo recursos que permitem a identificação da autoria, assim como das alterações/contribuições dos demais membros que fazem parte do grupo. Dessa maneira, pode-se reconhecer e identificar o processo de construção de cada um, além de proporcionar condições para a autonomia participativa e a autoria dos sujeitos integrantes do processo.

3. As Funcionalidades do *E*ditor de *T*exto *C*oletivo

O princípio de funcionamento do ETC é a inserção de parágrafos (texto ou imagem) por diferentes sujeitos, de forma não linear. Cada parágrafo apresenta um quadro com os dados de quem o postou pela primeira vez, data e hora da última alteração, como pode ser visto na figura 1. Resumidamente, o ETC funciona da seguinte forma: o usuário acessa a URL, cadastra-se, loga-se ao ambiente e, automaticamente, participa da comunidade padrão. A partir daí ele pode criar textos e participar de outros já criados, através de uma solicitação pela própria ferramenta ao participante responsável.

Figura 1 - Tela para visualização dos parágrafos no ETC

Nestes, além da inserção de parágrafos, há a possibilidade de edição e exclusão dos mesmos, bem como, a inserção de comentários. Este último não faz parte do corpo do texto, mas têm extrema importância, porque permite a discussão de uma idéia na própria ferramenta. Cabe salientar, que todas as interações entre o usuário e o texto ficam registradas e disponíveis para serem visualizadas através do item histórico. Isso evita que algum usuário mal intencionado altere ou exclua a contribuição do colega e permaneça no anonimato.

O ETC, na sua versão atual, apresenta, além das funcionalidades já citadas, a possibilidade de organização dos usuários em comunidades e o download da versão final a qualquer momento. As comunidades são maneiras de formar grupos dentro de uma mesma plataforma, de forma a organizar os textos e suas temáticas, evitando que na tela inicial apareça uma extensa lista de todos os textos criados no ETC. Qualquer usuário pode criar uma nova comunidade e inserir os demais participantes na mesma. Já o download permite que o texto possa ser transferido para a máquina local, nos formatos RTF, PDF e HTML, para que seja editado como o usuário desejar.

4. Experiências realizadas

O ETC já foi utilizado por alunos de Ensino Fundamental (3ª e 4ª série), Ensino Médio (3o ano), Cursos de Graduação e Pós-graduação.

Na 4a série, por exemplo, o ETC foi utilizado com o intuito de construir textos coletivos, ora de forma síncrona, ora de forma assíncrona. Cabe salientar que o ETC foi utilizado em dois momentos formais de aula e, as demais interações ocorreram fora deste espaço, por interesse dos próprios alunos.

O ETC também foi usado durante as aulas de Física do 3o ano do curso ROODA Tekton (Moresco, 2003), para elaborar textos cooperativos sobre os assuntos que estavam sendo discutidos em aula. Através destes textos pode-se acompanhar o processo de tomada de consciência de alguns conceitos físicos, entre eles, a aceleração centrípeta, atrito, velocidade, vetor velocidade, impulso, quantidade de movimento, que estavam sendo estudados pelos sujeitos da pesquisa

A partir das experiências com os alunos das disciplina EDU3375 - O Computador na Educação[1] (02/2002) e de Ambientes de Aprendizagem Computacionais[2] (01/2002) foram coletados os dados e realizado um levantamento dos erros e problemas de interface do ETC, levando em conta os relatos dos alunos/professores, com o fim de aperfeiçoas a ferramenta.

5. Trabalhos atuais e futuros

Atualmente, está se iniciando o processo de integração do ETC à plataforma ROODA[3], de modo que ele seja acessado através do ambiente, utilizando os recursos do mesmo e restrito aos usuários cadastrados no sistema. Além disso, sentiu-se a necessidade de desenvolver uma opção de edição dos parágrafos off-line, já que muitas pessoas acessam a Internet através da linha telefônica. Assim, pretende-se desenvolver um software cliente. O usuário, no momento em que se inclui no texto coletivo, pode baixá-lo em seu próprio computador, editando-o mesmo sem estar conectado à rede. Quando ele entrar na Internet, o software automaticamente enviará as alterações que foram realizadas para o servidor, executando o download das modificações dos demais usuários. Essa restrição da versão atual do ETC impediu, por exemplo, que o presente artigo fosse escrito a partir da ferramenta, já que nem todos os autores possuíam acesso de banda larga à Internet. Isso ocasionou um grande problema de controle das versões que circulavam entre os autores. Para realizar essas modificações, a equipe do NUTED pretende utilizar ferramentas já conhecidas na comunidade do Software Livre, tais como o CVS e o TEX. Outro aspecto que faz parte das atenções da equipe de desenvolvimento do NUTED encontra-se na interface do ETC, em vista dos estudos já realizados (Behar, 2002b).

6. Referências Bibliográficas

BAKHTIN, M. (2000) *A Estética da Criação Verbal*. 3ª ed. São Paulo: Martins Fontes.

BEHAR, P.; BITTENCOURT, J. ; BRAUOWERS, M.; KIST, S. (2002a) *ROODA DEVEL: Uma proposta de Framework para a Construção de Plataforma de Educação à Distância. In:* Workshop Software Livre 2002.

BEHAR, P.; BITTENCOURT, J. ; KIST, S.; PINHEIRO, A. (2002b) *Un estudio de interface para el ambiente ROODA - Rede cOOperativa De Aprendizagem. In:* VI Congreso Iberoamericano de Informática Educativa, Vigo, Espanha, nov. 2002

1 Esta disciplina é oferecida pelo curso de Pedagogia da Universidade Federal do Rio Grande do Sul.

2 Esta disciplina é oferecida pelo curso de Pós- Graduação em Educação e pelo Pós- Graduação em Informática na Educação da Universidade Federal do Rio Grande do Sul.

3 Rede Cooperativa de Aprendizagem – plataforma de educação à distância projetada pelo NUTED e disponível em http://rooda.edu.ufrgs.br.

Software Livre nas Pequenas e Médias Empresas: Tecnologia e Oportunidades

Douglas Kellermann[1], Dilney Albornoz Gonçalves[2], Débora Nice Ferrari[3]

[1]Centro Universitário Feevale (FEEVALE)
Rodovia RS 239, 2755 – 93.352-000 – Novo Hamburgo – RS – Brasil

[2] PPGA - Escola de Administração – Universidade Federal do Rio Grande do Sul (UFRGS)
Rua Anita Garibaldi, 372 – 90.450-000 – Porto Alegre – RS – Brasil

[3]Centro Universitário La Salle (UNILASALLE)
Av. Victor Barreto, 2.288 – 92.010-000 – Canoas – RS – Brasil

douglas@feevale.br, dilneyag@terra.com.br, nice@lasalle.tche.br

Abstract. *The free software initiative spread through Internet a set of new software where GNU/Linux is the most meaningful and know product. These software have been attracted a growing number of users, due to its characteristics as such as freedom to use and no costs. Among these new users, small and medium enterprises can be highlighted. Thus, this text aims to present a guide to free software use in small and medium enterprises.*

Resumo. *O movimento software livre fez proliferar, através da internet, um grande conjunto de software, sendo o GNU/Linux seu produto mais significativo e conhecido. Através de características como a livre utilização e o custo zero de licenciamento, estes softwares vêm atraindo um número de usuários cada vez maior. Dentre estes usuários, destacam-se as pequenas e médias empresas. Sendo assim, este texto tem como objetivo geral apresentar um referencial para a utilização de software livre nas pequenas e médias empresas.*

1. Introdução

A Tecnologia da Informação (TI), representa o agrupamento de recursos de computação, telecomunicação e multimídia, para as mais diversas aplicações, através da combinação de componentes de hardware e software. Assim a TI se torna, cada vez, indispensável para as empresas, em especial para as de pequeno e médio porte, tanto para atividades administrativas como na gestão do processo produtivo. Neste sentido, contar com uma estrutura de TI sedimentada sobre uma infra-estrutura robusta e flexível, torna-se fundamental para o obter maior proveito dos recursos computacionais [Kellermann, 2002].

A gratuidade de licenciamento é um dos mais fortes argumentos na adoção de software livre [Software Livre, 2003], principalmente no meio empresarial, o qual sofre com a crescente pressão dos órgãos de combate à pirataria de software. Neste contexto, as pequenas e médias empresas têm a oportunidade de adquirir e/ou incrementar a estrutura de TI para o seu negócio, com menor investimento, pela gratuidade de licenciamento e, assim, obter melhoria em seus processos pelo uso da informática.

Para que possa haver a compreensão completa do contexto envolvendo a adoção de tecnologia de software, baseadas nos modelos de Software Livre e *Open Source*, pelas pequenas e

médias empresas, é necessário conhecer as necessidades de TI das mesmas, avaliar como o software livre pode atender estas demandas e ainda compreender de que forma estas empresas vêem a adoção de software livre. Sendo assim, este artigo combina os resultados de dois importantes trabalhos de conclusão de curso sobre software livre e seu uso em pequenas e médias empresas [Kellermann, 2002], [Gonçalves, 2003].

Este texto está organizado em 5 partes. A seção 2 deste artigo aborda a forma que a TI é empregada nestas empresas, a seção 3 sugere alguns softwares livres para o atendimento das suas necessidades e a quarta seção destaca oportunidades relacionadas à utilização destes softwares. Finalmente, na seção 5 as conclusões do trabalho são apresentadas.

2. Tecnologia da Informação nas Pequenas e Médias Empresas

Na necessidade de avaliar a utilização dos diferentes tipos de softwares pelas pequenas e médias empresas, Kellermann realizou uma pesquisa junto às empresas da Região da Grande Porto Alegre. Esta pesquisa teve por finalidade identificar a existência de opções de TI em software livre, equivalente às opções não livres, para os recursos utilizados nas empresas. Participaram 126 empresas com acesso à Internet, visto que era necessário o preenchimento *on line* da pesquisa ou envio do formulário através de *e-mail*. Com o resultado da pesquisa, foi possível identificar o perfil de uso da TI nas pequenas e médias empresas desta região geográfica. Os dados mais importantes obtidos estão destacados na Tabela 1 abaixo.

Tabela 1. Aspectos de TI em pequenas e médias empresas

Tecnologias Empregadas	Percentual
Estrutura com até 10 computadores (com e/ou sem rede)	67.5%
Com acesso a internet (ADSL e/ou Discada)	64.4%
Ferramentas tipo *Office* (Processador de texto, Planilha eletrônica e Banco de dados)	83,0%
Outras ferramentas *desktop* (Gráficos, Internet e Agenda)	69,3%
Servidor (Arquivos, Banco de dados e Intranet)	37,5%
Aplicativos administrativos (Financeiro, Pedidos/Faturamento e Governo (Rais/Sefip/IR))	48,1%
Intercâmbio de informações (Troca de arquivos/Acesso a serviços on-line)	68,3%
Disponibilidade de equipe técnica interna (hardware e software)	20,1%

Para realização da pesquisa, foi desenvolvido um formulário *web* (usando recursos de software livre) onde as empresas respondiam as questões relacionadas à pesquisa. Este site está disponível no endereço http://slpme.codigolivre.org.br, assim como todo trabalho desenvolvido por Kellermann em [Kellermann, 2002].

3. Software Livre para as demandas tecnológicas nas Pequenas e Médias empresas

O GNU/Linux disseminou amplamente o conceito de Software Livre e *Open Source*. Os softwares produzidos nestes modelos de desenvolvimento, como o MySQL, o Apache, Gnome/KDE, dentre outros, já podem atender uma grande variedade de demandas, sendo a infra-estrutura seu ponto forte. Também, o número de aplicações *desktop* vem apresentando contínuo crescimento [Kellermann 2002].

Nas pequenas e médias empresas o uso de software livre está cada vez mais presente. A pesquisa realizada por Kellermann apresenta aspectos interessantes quanto ao conhecimento de software livre neste segmento. Os resultados mostram que as empresas têm alguma informação sobre software livre, aliado ao interesse de conhecer mais sobre o assunto. Tem maior importância para estas empresas a economia que o emprego desta tecnologia pode trazer através do licenciamento gratuito. Esta economia, muitas vezes, é revertida em investimentos na própria empresa. A parcela de empresas que utiliza esta tecnologia, o faz, em ambiente servidor e *desktop*.

A Tabela 2 apresenta as informações que as empresas possuem do software livre e como estes são utilizados.

Tabela 2. Pequenas e Médias empresas e o Software Livre

QUESTÕES RELACIONADAS	RESULTADO OBTIDO	
Conhecimento sobre software livre	Tem alguma informação	44,37%
	Querem saber sobre	21,8%
Qual seu pensamento sobre software livre	É interessante pelo licenciamento gratuito	36,2%
	Não tem informação suficiente para opinar	16,4%
	É interessante pela qualidade	13,5%
Sobre a utilização de software livre	Não utilizamos	40,9%
	Utiliza na estação	14,2%
	Utiliza no servidor	13,0%

Por sua vez, Gonçalves [Gonçalves, 2003] conduziu uma importante pesquisa relacionada a aceitabilidade do software livre nas empresas. Nesta pesquisa, diferentes aspectos da utilização de software livre nas pequenas e médias empresas foram avaliados, resultando nos seguintes pontos conclusivos:

- A dispersão das opiniões indica que o software livre ainda não é uma unanimidade;
- A redução de custos e a independência de fornecedor surgem como as maiores vantagens;
- Os entrevistados que já utilizam software livre acreditam que este tipo de software atende as suas necessidades;
- Existe boa aceitabilidade quanto à utilização de software livre.

Com base no perfil apresentado na Tabela 1 e na visão e conhecimento quanto ao software livre que as empresas apresentam (Tabela 2), Kellermann apresenta algumas ferramentas em software livre segmentadas por aplicação, conforme demonstra a Tabela 3. Esta não representa uma referência definitiva a respeito de disponibilidade de software livre para as respectivas aplicações, mas uma indicação básica de recursos disponíveis. O ponto forte é oferta de recursos para infra-estrutura [Kellermann, 2002].

Tabela 3. Algumas Opções de Software por Aplicação

SERVIDOR	
Servidor (GNU/Linux)	Debian, Slackware, Suse, Conectiva entre outros
Servidor Arquivos/Impressão	Samba e NFS
Servidor Http/Firewal	Apache (PHP/outros módulos) e Squid/IPTables/Snort
Servidor Banco de dados	MySQL, PostgreSQL e FireBird
Emuladores	DosEmu, FreeDos, Samba e Wine
Serviço de Diretório	OpenLDAP
DESKTOP	
Ferramentas tipo office	OpenOffice/Gnome-Office/Koffice
Editores Gráficos (Raster/Vetorial)	Gimp e Sketch
Internet (Browser/Correio)	Mozilla, Konqueror/Kmail
Colaboração (Agenda)	Direto GNU/moregroupware
Administrativo	ERP Sistema Processa/ERP Multi-Informática/Compiere ERP

Vale destacar que estes software apresentam uma série de vantagens em relação à adoção de produtos de software proprietário como Microsoft, Oracle, Novell, Borland, CA e outros [Kellermann, 2002]. A qualidade, principal característica técnica, é superior aos softwares proprietários pela maior revisão do código fonte. A disponibilidade do código fonte também proporciona maior segurança e desempenho ao software, aliado ao fato de que uma estrutura UNIX permite estabilidade, robustez e escalabilidade superiores [Kellermann, 2002].

4. Software Livre como gerador de oportunidades

Não diferente de outras plataformas, a adoção de software livre requer que serviços relacionados, como implantação e migração, sejam avaliados e conduzidos por profissionais capacitados, de forma a alcançar, de maneira mais segura, os resultados esperados. Sobre tudo, processos de migração necessitam atenção especial por se tratar da mudança de uma prática já estabelecida, principalmente se observados os índices de utilização de aplicativos administrativos, que figuram como "software legado" em processos de migração.

Por isto, se faz necessária a atuação de pessoal especializado para orientação/condução dos trabalhos. Este nicho conta com a crescente, mas ainda reduzida, oferta de empresas e profissionais qualificados. A rica e crescente oferta de software livre, por si só, não representa a garantia de sucesso em processos de implantação e/ou migração de tecnologia nas empresas. É fundamental o estudo de viabilidade, por profissionais da própria empresa ou terceiros, para determinar como a implantação ou migração deverá ser conduzida. Nestes processos, a solução mista, combinando recursos de software livre e software proprietário, ainda é comum e necessária. Isto se deve em função da necessidade de utilizar/manter softwares proprietários legados (ou dependentes de plataforma proprietárias), ou ainda, sem equivalente em software livre.

Além dos serviços baseados em consultoria, treinamento, implantação de produtos de infra-estrutura já consolidados como o GNU/Linux, Apache, PostgreSQL/MySQL, entre outros; serviços para desenvolvimento de software administrativo, no modelo de software livre, tem encontrado espaço. Este fato é identificado através da pesquisa conduzida por Kellermann com relação ao uso de software administrativo pelas micro e pequenas empresas. Como resultado desta pesquisa, aplicações como Administração de Pessoal, Contabilidade e Custos e Controle Fiscal surgiram como necessidade potencial em 45% das empresas [Kellermann, 2002].

5. Conclusão

Este trabalho apresentou um referencial para a utilização de software livre nas pequenas e médias empresas. Concluiu-se que a utilização de software livre gera um grande conjunto de benefícios tecnológicos e econômicos. A disponibilidade do código fonte permite que as empresas possam se beneficiar da independência de fornecedores, item estratégico importante percebido pelas empresas, assim como permite a auditoria e a revisão extrema do código, propiciando confiabilidade ao software. Isto tudo, ainda fomenta o desenvolvimento tecnológico, pelo livre acesso aos fontes deste software. A demanda de serviço gera um relacionamento comercial local, entre as empresas que utilizam e as empresas que oferecem serviços de tecnologia, gerando renda para aplicação e desenvolvimento local.

A convergência dos trabalhos de Kellermann e Gonçalves demonstra que o assunto Software Livre nas pequenas e médias empresas não é mais somente alvo de pesquisa das Ciências ligadas à Computação. Também os administradores percebem a oportunidade mútua do emprego empresarial desta tecnologia, sobre tudo nas pequenas e médias empresas.

Referências

Gonçalves, Dilney Albornoz. (2003) “Pesquisa de Aceitabilidade do Software Livre nas empresas do Rio Grande do Sul”, PPGA - Escola de Administração, UFRGS.Abril.(Trabalho de Conclusão de curso).

Kellermann, Douglas. (2002) “Software Livre na Infra-Estrutura de Tecnologia da Informação da Pequena e Média Empresa”, Curso de Ciência da Computação, FEEVALE, Dezembro. (Trabalho de Conclusão de Curso).

Projeto Software Livre. (2003). http://www.softwarelivre.org.

Software Livre na Micro e Pequena Empresa. (2003). http://slpme.codigolivre.org.br

Software Livre como Alternativa a Processos de Apuração de Dados em ONGs Prestadoras de Serviços[1]

Ítalo Flexa Di Paolo

Bacharel em Ciência da Computação – Centro Universitário do Estado do Pará (CESUPA)
Discente de Engenharia Elétrica – Universidade Federal do Pará (UFPA)

flexa@canal13.com.br

Abstract. *The non-profit Organizations services renderers, that already find difficulties to achieve its activities purposes, find hard resistance to computerization due to its cost. This work shows how free software can offer good alternatives for these organizations through a case study made in the refine process of a field research, offering, this way, more resources for the decision making and implementation of public policies.*

Resumo. *As Organizações Não-Governamentais prestadoras de serviços, que já encontram dificuldades para realizar suas atividades fins, encontram forte resistência à informatização devido ao seu custo. Este trabalho demonstra como os software livres podem proporcionar boas alternativas para estas organizações através de um estudo de caso feito no processo de apuração de dados de pesquisa de campo, proporcionando, assim, maiores recursos para a tomada de decisão e implementação de políticas públicas.*

1. Introdução

Organizações Não-Governamentais (ONGs) prestadoras de serviços são aquelas que desenvolvem serviços que atendam as necessidades humanas básicas, como educação, saúde, habitação e bem-estar social em geral (KISIL[2], [199-]). Grande parte destas organizações é sustentada através de doações feitas pela sociedade ou pelo governo, fazendo com que pouca verba seja investida em tecnologias de informação.

Neste tipo de organização, além dos serviços prestados a sociedade, algumas também desenvolvem pesquisas de campo, tanto com interesse acadêmico como visando outros interesses em prol da implementação de políticas públicas. Pela falta de tecnologias de informação viáveis e eficientes, normalmente o processo de apuração e análise dos dados destas pesquisas é feito de forma manual, o que provoca inúmeras desvantagens como maior dispêndio de tempo e pouca flexibilidade para a análise dos dados.

Visando oferecer uma tecnologia acessível a este tipo de organização, foi desenvolvido um Sistema de Informação (SI) para atuar como facilitador dos processos de apuração e análise dos dados de pesquisa, dando maior subsídio para estudar e conhecer melhor a realidade, reconhecer seus problemas e avaliar evoluções, assim como apontar possíveis soluções a problemáticas específicas. Os *software* existentes hoje no mercado para este tipo de finalidade são de altos custos e com poucas possibilidades de adaptações a projetos específicos. Diante desta realidade, as tecnologias de distribuição livre são alternativas ideais.

O sistema desenvolvido, denominado SIAD (Sistema de Informação para Apuração de Dados), foi testado e validado pela ONG Centro Nova Vida[3]. Neste trabalho são discutidas suas principais características de modelagem, de implementação, bem como os resultados com ele obtidos.

---

[1] Este artigo foi desenvolvido a partir do Trabalho e Conclusão de Curso de Ciência da Computação, do mesmo autor, orientado pela Profª MSc. Andréa Cristina Marques de Araújo, CESUPA: Belém, 2002.
[2] IOSCHPE, Evelyn Berg (Org.). *3º Setor: desenvolvimento social sustentado*. 2ª Ed. Rio de Janeiro: Paz e Terra, 2000.
[3] Com sede em Ananindeua-PA (http://www.centronovavida.org.br), o Centro Nova Vida atua na reabilitação de portadores de dependência química (tóxicos e álcool) na região metropolitana de Belém-PA.

2. Modelagem e Desenvolvimento do SIAD

O SIAD foi desenvolvido através do *Rational Unified Process* (RUP), descrito por FURLAN (1998), cujos pontos principais são discutidos nos itens seguintes.

2.1. Requisitos

Uma das vertentes de trabalho do Centro Nova Vida é a pesquisa. Como saber quais localidades possuem grandes concentrações de dependentes químicos? Qual o perfil destes portadores? Qual a metodologia mais adequada? Enfim, são tantas as perguntas e um dos principais meios de se obter respostas é a pesquisa de campo. O SIAD tem o objetivo de sistematizar o processo de apuração e análise dos dados destas pesquisas, feitas através da aplicação de formulários (e/ou questionários) de perguntas fechadas, diminuindo o tempo e os custos destas pesquisas, e fornecendo uma ferramenta poderosa de consulta.

Embora o SIAD tenha sido baseado num estudo de caso em uma ONG específica, ele se propõe a ser um modelo em que possam ser cadastrados diversos tipos de formulários e/ou questionários, tornando a consulta o mais flexível possível, podendo fazer o cruzamento entre diversas perguntas ou respostas, seja qual for o número de formulários cadastrados.

2.2. Análise e Projeto

Para atingir os objetivos propostos, os formulários foram modelados contendo apenas perguntas fechadas, podendo ser multi ou mono-valoradas. Cada formulário foi dividido em três partes, sendo uma a sua identificação, composta por campos com valores descritivos, como data e nome do entrevistado, e campos com valores definidos, como os nomes dos entrevistadores, as perguntas e as opções de resposta.

Um detalhe importante é que para cada pesquisa, um formulário deverá ser cadastrado contendo a estrutura descrita acima, sendo gerado para ela um banco de dados separado com as tabelas e relacionamentos necessários. Cada base de dados será referenciada em uma base principal em que estarão contidas também as suas descrições, para que o pesquisador possa ter acesso a cada uma delas.

No caso do Centro Nova Vida, foram detectados dois atores para o sistema, sendo um pesquisador, o qual poderá ter acesso a todas as pesquisas, e um digitador que terá acesso apenas a determinadas pesquisas com permissão apenas de cadastrar os formulários aplicados no campo. O Diagrama 1 mostra a essência da estrutura de classes proposta para cada pesquisa.

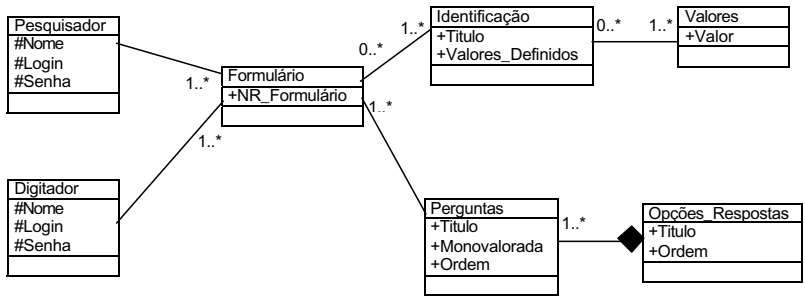


Diagrama 1: Relacionamento entre as Classes do SIAD

2.3. Implementação

Diante da possibilidade de vários pesquisadores consultarem o sistema ao mesmo tempo, e da flexibilidade e portabilidade que o modelo de quatro camadas possui, o ambiente *web* foi escolhido. Foram instalados o servidor Apache (APACHE…, 2002) e o banco de dados MySQL (MYSQL…, 2002), e a linguagem escolhida foi a PHP (PHP…, 2002).

O banco de dados MySQL também fornece recursos muito simples para a gerência das bases de dados dinâmicas, necessárias para cada pesquisa. No Diagrama 2 pode ser observado o modelo de dados proposto com as principais estruturas para cada uma das bases. Como se pode perceber, na base de cada pesquisa existe uma tabela principal, chamada *Formulário*; por se tratar de uma tabela dinâmica, seus campos que guarda os valores das identificações (nos campos *Identificação?*) e das respostas mono-valoradas (campos *Pergunta?*) onde o símbolo "*?*" remete ao código do campo *CD_Identificação*, na tabela *Identificação*, e *CD_Respostas*, da tabela *Opções_Respostas*. Por exemplo, a identificação "Data da Entrevista" é registrada na tabela *Identificação* com *CD_Identificação* igual a *4*; assim sendo, o campo Identificação4 guarda os valores desta identificação em cada um dos formulários.

Do mesmo modo acontece com as perguntas multi-valoradas, cujas respostas são armazenadas no campo *Alternativa* da tabela *Pergunta?*. Deste modo é possível criar qualquer formulário em tempo de execução.

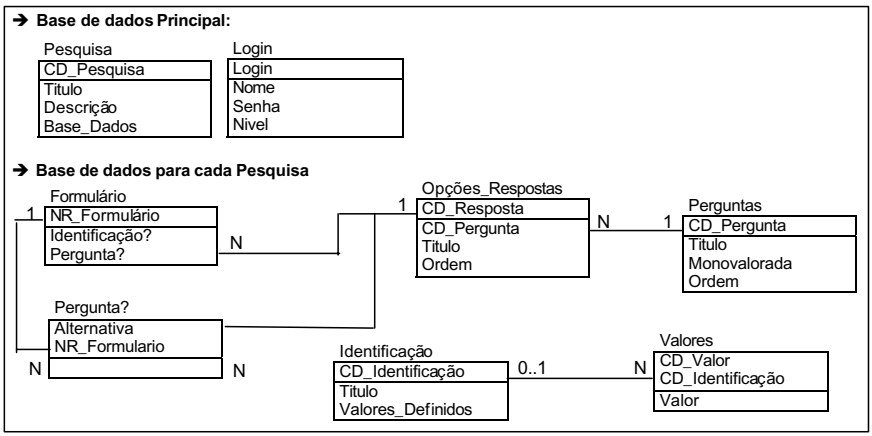


Diagrama 2: Modelo de dados proposto para o SIAD

Diversos tipos de consultas foram implementados, como as freqüências de cada resposta, o cruzamento (ver Figura 1) entre diversas perguntas ou entre diversas respostas diferentes, verificação dos dados de determinados formulários, perguntas não respondidas, entre outras, dando abertura para que outras novas possam ser codificadas de acordo com necessidades específicas. Assim, é possível executar diversos tipos de consulta, proporcionando ao pesquisador enormes facilidades e melhores subsídios na análise dos dados da pesquisa.

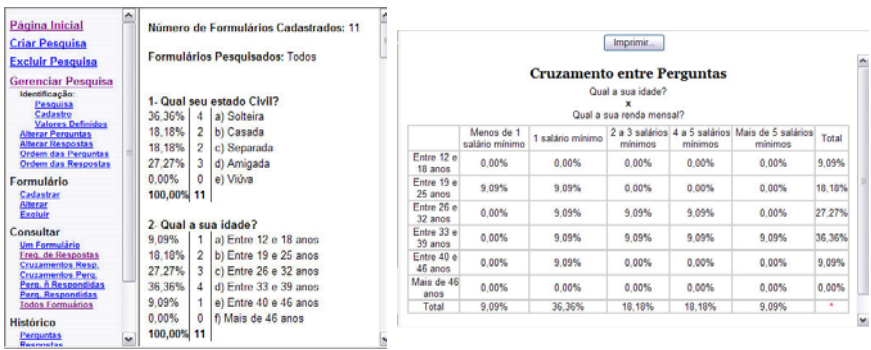

Página Inicial
Criar Pesquisa
Excluir Pesquisa
Gerenciar Pesquisa
Formulário
Consultar
Histórico
Número de Formulários Cadastrados: 11
Formulários Pesquisados: Todos
1- Qual seu estado Civil?
36,36% 4 a) Solteira
18,18% 2 b) Casada
18,18% 2 c) Separada
27,27% 3 d) Amigada
0,00% 0 e) Viúva
100,00% 11
2- Qual a sua idade?
9,09% 1 a) Entre 12 e 18 anos
18,18% 2 b) Entre 19 e 25 anos
27,27% 3 c) Entre 26 e 32 anos
36,36% 4 d) Entre 33 e 39 anos
9,09% 1 e) Entre 40 e 46 anos
0,00% 0 f) Mais de 46 anos
100,00% 11
Imprimir...
Cruzamento entre Perguntas
Qual a sua idade?
x
Qual a sua renda mensal?
Menos de 1 salário mínimo
1 salário mínimo
2 a 3 salários mínimos
4 a 5 salários mínimos
Mais de 5 salários mínimos
Total
Entre 12 e 18 anos 0,00% 0,00% 0,00% 0,00% 0,00% 9,09%
Entre 19 e 25 anos 9,09% 9,09% 0,00% 0,00% 0,00% 18,18%
Entre 26 e 32 anos 0,00% 9,09% 9,09% 9,09% 0,00% 27,27%
Entre 33 e 39 anos 0,00% 9,09% 9,09% 9,09% 9,09% 36,36%
Entre 40 e 46 anos 0,00% 9,09% 0,00% 0,00% 0,00% 9,09%
Mais de 46 anos 0,00% 0,00% 0,00% 0,00% 0,00% 0,00%
Total 9,09% 36,36% 18,18% 18,18% 9,09%

Figura 1: Algumas das formas de consulta oferecidas pelo SIAD

3. Conclusão

Neste trabalho, o conceito de Sistema de Informação foi desenvolvido com o objetivo principal de dinamizar o processo de apuração de dados de pesquisa de campo, como auxílio à análise de dados da pesquisa. E para atingir os objetivos, o SIAD foi desenvolvido com ferramentas de distribuição livre, não exigindo ônus algum, como por exemplo, com licenças de uso, fornecendo não somente uma ferramenta de auxílio, mas também uma alternativa viável à realidade de Organizações Não Governamentais, como o Centro Nova Vida.

Com o SIAD, a Tecnologia da Informação tornou-se algo acessível ao Centro Nova Vida, transformando o processo de apuração, antes totalmente manual e trabalhoso, exigindo bastante tempo para concluí-lo, em passos simples e mais confiáveis, podendo o usuário gerenciar diversas pesquisas e visualizar diversos tipos de consultas a qualquer momento.

Com este processo, testado em três pesquisas diferentes (duas com cerca de 80 formulários com 70 questões cada e uma com cerca de 300 formulários de 80 questões cada), o tempo necessário para realizar a apuração dos dados diminuiu em mais de 90%, facilitando em muito a análise da pesquisa, oferecendo maior eficiência e recursos, e possibilitando diversos tipos de consulta executadas em tempos desprezíveis, e a custo zero.

A partir deste estudo de caso, o SIAD pode apresentar-se como uma ferramenta simples e prática para auxiliar a análise de pesquisas de campo e, por ser multi-plataforma e multi-usuário, é extremamente útil, principalmente para organizações sem fins lucrativos ou trabalhos de pesquisa em diversas áreas da ciência.

4. Referências

APACHE, Version 2.0.43. Desenvolvido por The Apache Software Foundation. [S.l.], 2002. Disponível em: <http://archive.cs.uu.nl/mirror/apache.org/dist/httpd/>. Acesso em: 05 mar. 2002.

ARAÚJO, Andréa Cristina Marques de. A informação como fator diferenciador para o sucesso estratégico das organizações. *A Revista da Informação e Tecnologia*, Campinas, ago. 2001. Disponível em: <http://www.revista.unicamp.br/infotec/artigos/andrea_cristina.html>. Acesso em: 02 mar. 2002.

CHIAVENATO, Idalberto. *Teoria Geral da Administração:* abordagens descritivas e explicativas. 5. ed. São Paulo: Makron Books, v. 2. 1998.

FURLAN, José Davi. *Modelagem de objetos através da UML:* the Unified Modeling Language. São Paulo: Makron Books, 1998.

IOSCHPE, Evelyn Berg (Org.). *3º Setor:* desenvolvimento social sustentado. 2. ed. Rio de Janeiro: Paz e Terra, 2000.

LAUDON, Jane Price; LAUDON Kenneth C. *Gerenciamento de Sistemas de Informação*. 3. ed. Rio de Janeiro: LTC, 2001.

MARCELLINO, Nelson Carvalho. Departamentalização e Unidade das Ciências Sociais. In: _____ (Org.). *Introdução às Ciências Sociais.* 3. ed. Campinas: Papirus, 1989. p. 103-110.

MYSQL, Version 3.23.49. Desenvolvido por The MySQL AB Company. [S.l.], 2002. Disponível em: <http://www.mysql.com/>. Acesso em: 05 mar. 2002.

O'BRIEN, James A. *Sistemas de Informação e as decisões gerenciais na Era da Internet.* São Paulo: Saraiva, 2002.

PHP, Version 4.2.1. Desenvolvido por The Apache Software Foundation. [S.l.], 2002. Disponível em: <http://www.php.net/>. Acesso em: 05 mar. 2002.

GNU/Linux em máquinas IPF: Promessa ou realidade?

Rodrigo Dias Cassali[1], **Marco Aurélio Stelmar Netto**[1], **Caroline Bellan Oliva**[1], **Augusto Mecking Caringi**[1], **César Augusto Fonticielha De Rose**[1]

[1]Centro de Pesquisa em Alto Desempenho*
Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga 6681, Prédio 30 - Sala 141
90619-900 Porto Alegre, RS

{cassali, stelmar, caringi}@cpad.pucrs.br, {coliva, derose}@inf.pucrs.br

Abstract. Given the daily increase of amount and complexity of services, servers require more processing power and data storage capacity. Guided by this idea, Intel and HP have been working together on a 64-bit architecture, named IA64, that originated the Itanium Processor Family (IPF). The open source developers have been following these trends and have already updated the GNU/Linux operating system for this architecture. This paper presents an overview about the Itanium processor, the GNU/Linux support for the IA64 architecture, porting applications and groups that foster the development of tools.

Resumo. Com a crescente quantidade e complexidade de serviços, os servidores precisam dispor cada vez mais de alto poder de processamento e grande capacidade de armazenamento de dados. Norteada por essa idéia, a Intel em parceria com a HP vem trabalhando em uma arquitetura de 64 bits, chamada IA64, que deu origem a Família de Processadores Itanium (IPF). Os desenvolvedores de software livre vêm acompanhando as novas tendências e já atualizaram o sistema operacional GNU/Linux para esta nova arquitetura. Este artigo apresenta uma visão geral do Itanium, além do suporte do GNU/Linux para a arquitetura IA64, porte de aplicações e movimentos de apoio ao desenvolvimento de ferramentas.

1. Introdução

Com o crescente número de usuários e aplicações GNU/Linux, e também a crescente necessidade de servidores que provêem maior poder computacional [Intel, 2002b], são necessárias arquiteturas que resolvam este problema. O Itanium apresenta características que fornecem uma solução para o problema apresentado. Com o Intel Itanium 2, a segunda geração do processador Itanium, o nível de poder computacional cresceu. A Intel estima que softwares escritos para processadores Itanium 1, executados em Itanium 2, ofereceram de 1.5 a 2 vezes mais desempenho sobre o processador Itanium 1 (sem recompilação de código). Além disso, o fato do GNU/Linux ser um sistema de código aberto, faz com que surjam novas aplicações e ferramentas de desenvolvimento para o sistema operacional.

Este artigo está organizado da seguinte forma: a seção 2 apresenta a diferença entre IA64 e Itanium; a seção 3 comenta sobre o suporte do GNU/Linux para arquitetura Itanium; a seção 4 ilustra alguns detalhes importantes a serem considerados durante o desenvolvimento e porte de aplicações; a seção 5 comenta sobre os projetos relacionados ao desenvolvimento para arquiteturas IA64; e a seção 6 apresenta as considerações finais.

*Projeto em cooperação com a HP Brasil.

2. Itanium e IA64

A arquitetura IA64 [Intel, 2002a] é nova, rompendo, dessa forma, com a arquitetura x86 (também conhecida como IA32) e baseando-se em um novo paradigma de projeto chamado EPIC, do inglês *Explicit Parallel Instruction Computing*, que é traduzido como Computação com Paralelismo de Instruções Explícito. Tal abordagem torna o compilador o grande responsável por determinar e explicitar o paralelismo presente nas instruções a serem executadas. De acordo com a Intel, EPIC deve ser visto como um novo paradigma na construção de processadores da mesma forma que existem RISC e CISC [Intel, 2003].

O EPIC é inspirado na técnica VLIW, que significa Palavra de Instrução Muito Grande, do inglês *Very Large Instruction Word*. Processadores que usam essa técnica acessam a memória transferindo longas palavras de programa, sendo que, em cada palavra estão empacotadas várias instruções [CLRC, 2003]. No caso da IA64, são usadas três instruções para cada pacote de 128 *bits*. Cada instrução possui 41 *bits* de tamanho, os 5 *bits* restantes são gerados pelo compilador, e utilizados para informar quais instruções podem ser executadas em paralelo.

Inicialmente conhecido como Merced, o primeiro processador baseado na arquitetura IA64 chegou ao mercado em 2001 e foi batizado pela Intel com o nome Itanium. Era um processor de 295 milhões de transistores, sendo que 270 milhões compunham a cache L3 e os outros 25 milhões o núcleo em si. Foi disponibilizado em 2 versões: 733 Mhz e 800 Mhz. Em 2002 foi lançado o Itanium 2, com uma série de melhorias sob o primeiro, maior memória cache e *clock* de 1 GHz [Intel, 2003]. Além do desempenho que é obtido através do alto grau de paralelismo interno no processador, o Itanium [Intel, 2002b], por ser uma arquitetura de 64 *bits*, também fornece como vantagens, suporte a uma quantidade muito maior de memória física, espaço de endereçamento virtual de 1 milhão de Terabytes, e registradores de ponto flutuante de 82 *bits* que possibilitam um desempenho maior em cálculos matemáticos.

Para facilitar a transição de 32 para 64 *bits*, foi adicionado ao Itanium um modo de compatibilidade x86, o que o torna capaz de rodar aplicações e sistemas operacionais de 32 *bits* sem qualquer modificação. Isto é feito através de um decodificador que traduz instruções x86 em instruções IA64, entretanto existe comprometimento do desempenho. Desta forma, para extrair ao máximo o poder do Itanium o ideal é recompilar o código para 64 *bits*.

3. Suporte e Desenvolvimento de Aplicações GNU/Linux para IPF

3.1. Kernel e Distribuições GNU/Linux

A primeira versão do *kernel* para o Itanium foi lançada em 2 de fevereiro de 2000 . Depois de otimizações e novas funcionalidades, a versão 2.4.0 do *kernel* foi liberada em 4 de janeiro de 2001. Esta foi a primeira versão estável do *kernel* a suportar processadores Itanium [kernel.org, 2003].

Para o porte do *kernel* à essa nova arquitetura, foi realizada uma divisão de tarefas entre as empresas interessadas, tais como: HP, Intel e VA Linux Systems. Outras empresas tais como IBM, Red Hat, SGI e CERN ficaram responsáveis por desenvolvimento de compiladorss, bibliotecas , ferramentas de analise de desempenho entre outras. A configuração do *kernel* é feita do mesmo modo que na arquitetura 32 *bits*, a única diferença é em relação ao compilador. Para gerar a imagem do *kernel* é preciso que se tenha um compilador que suporte 64 *bits*, além da *glibc* e do pacote *binutils*.

Dentre as distribuições que suportam a arquitetura Itanium estão: Debian, Suse, Turbolinux, Red Hat, Caldera e Mandrake.

3.2. Desenvolvimento de Aplicações

Arquiteturas 64 *bits* diferem das arquiteturas 32 *bits* pelo fato de poderem endereçar maiores áreas de memória, utilizando uma palavra de 64 *bits* para endereçamento e 64 *bits* para dados, o que interfere na portabilidade de códigos. A Figura 1 mostra a forma errada e correta (nesta ordem) de se alocar N *bytes* em um código portável, em relação as diferenças de tamanhos dos tipos básicos [Coutant, 2000]. Essas diferenças devem ser observadas pelo fato de alguns programadores usarem, por exemplo, 4 *bytes* para alocar uma variável do tipo *char** em arquiteturas 32 *bits*, ao invés de usarem a função *sizeof()* para obter o tamanho real do tipo a ser alocado.

```
char **p;
p = malloc(4 * N);
```

```
char **p;
p = malloc(sizeof(char) * N);
```

Figura 1: Exemplos de código para alocação de memória em C.

Quando se quer compilar códigos para serem executados em uma arquitetura a qual não se tem acesso, pode-se utilizar um recurso chamado *Cross-Compiling*. Para tanto, é necessário que se tenha um compilador que suporte geração de código para arquiteturas de processadores diferentes do computador *host*. O compilador C/C++ do projeto GNU [GCC, 2003], por exemplo, suporta diversas plataformas *host* e processadores alvos. Isto pode ser desejado quando se quer compilar uma aplicação para outras arquiteturas, não precisando adquirí-las. Outro motivo bastante comum, é quando se tem a arquitetura, mas nela não se encontra um compilador nativo para tal, por exemplo, em sistemas embarcados (roteadores, telefones celular, impressoras, etc.) [Barr, 1999]. Outra situação na qual se utiliza *Cross-Compiling* é no caso de se ter vários desenvolvedores trabalhando com máquinas de uma arquitetura diferente da qual a aplicação é destinada. Por exemplo, 20 programadores desenvolvendo em máquinas Pentium IV e com apenas uma máquina Itanium para teste, a compilação do código é toda feita nos Pentium IV, apenas sendo necessário o teste da mesma no computador Itanium.

3.3. Movimentos de Apoio ao Desenvolvimento

O grupo Gelato [Gelato, 2003], fundado pela HP e outras sete instituições mundiais de pesquisa, tem o intuito de facilitar o avanço de estudos de áreas em desenvolvimento tais como ciências naturais. O Gelato promove o desenvolvimento de aplicações para a plataforma Itanium GNU/Linux, fornecendo informações que tornem esta plataforma mais acessível aos pesquisadores. A comunidade tem acesso aos softwares desenvolvidos pelo Gelato, bem como outros provenientes de comunidades contribuintes. As instituições interessadas em contribuir para os objetivos e influenciar nas decisões do grupo podem optar por tornarem-se membros ou patrocinadores do Gelato. Dentre os membros do grupo estão instituições de ensino e empresas de tecnologia que estão interessados em ferramentas GNU/Linux para plataforma Itanium.

Outro movimento, chamado Trillian [Trillian, 2003], iniciado em maio de 1999, com o objetivo de portar o Linux para IA64 e disponibilizá-lo para a comunidade antes ou juntamente com o lançamento do Itanium. As empresas que participam do projeto são: Caldera, CERN, Hewlett Packard, IBM, Intel, Red Hat, SGI, SuSE, Turbo Linux e VA Linux Systems. A comunidade software livre pode contribuir para o projeto adicionando funcionalidades às aplicações existentes, portando aplicações, otimizando o GCC [GCC, 2003] para IA64, portando drivers, etc.

4. Considerações Finais

Muito antes da introdução da primeira versão da família de processadores de 64 *bits* da Intel, os desenvolvedores de sistemas operacionais já estavam trabalhando na adaptação de seus sistemas para essa nova arquitetura. No mundo do software livre não foi diferente, e desde então vários grupos vêm trabalhando na adaptação do GNU/Linux para processadores de 64 *bits*.

Como vimos neste artigo, a maioria das distribuições GNU/Linux já suportam a arquitetura Itanium e a lista cresce rapidamente. O desenvolvimento de aplicações 64 *bits* já é suportado tanto a partir de plataformas de 32 *bits* (*Cross-Compiling*), como em ambientes 64 *bits* nativos e o número e a qualidade dos ambientes cresce à medida que a base instalada de máquinas IPF aumenta.

Com a questão do suporte do GNU/Linux já encaminhada, e podendo hoje ser considerada realidade, o grande problema parece ainda ser onde se justifica o investimento em uma arquitetura de 64 *bits* através de uma análise detalhada da relação custo benefício [Willard and Kaufmann, 2001], independentemente do sistema operacional utilizado.

Referências

Barr, M. (1999). *Programming Embedded Systems in C and C++*. O'Reilly & Associates Inc., 101 Morris Street, Sebastopol, CA 95472, first edition.

CLRC (2003). Processadores Intel Itanium IA64.
Disponível em http://www.cse.clrc.ac.uk/arc/ia64.shtml. Acessado em maio 2003.

Coutant, C. (2000). 64-Bit Application Development for PA-RISC & IA-64.

GCC (2003). Sítio do projeto gnu c compiler - GCC.
Disponível em http://gcc.gnu.org/. Acessado em maio 2003.

Gelato (2003).
Disponível em http://www.gelato.org. Acessado em maio 2003.

Intel (2002a). Intel Itanium 2 Processor Reference Manual.

Intel (2002b). The Intel Itanium 2 Processor: Uniquely Architected for Demanding Enterprise and Technical Applications.

Intel (2003). Sítio da família do Itanium .
Disponível em http://www.intel.com/design/itanium/family/. Acessado em maio 2003.

kernel.org (2003). The Linux Kernel Archives.
Disponível em http://www.kernel.org/. Acessado em maio 2003.

Trillian (2003). Trillian Project.
Disponível em http://www.linuxia64.org/pressfinal.pdf. Acessado em maio 2003.

Willard, C. G. and Kaufmann, N. (2001). Itanium Enters the High-Performance Computing Universe.

Gerenciamento de Processos em Controladores Programáveis Usando XML

Giovani M. Cascaes, João C. Netto

Instituto de Informática - Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RG – Brasil

{gmcascaes,netto}@inf.ufrgs.br

Resumo. *Os controladores programáveis tornaram-se fator decisivo para o controle de processos em ambientes industriais. Permitir o gerenciamento desses, eleva-os ao mesmo grupo de outros equipamentos da rede. Isso é possível através da linguagem XML, atualmente em ampla ascensão e aceitação. Este artigo apresenta as características dos controladores envolvidas no controle do processo, expõe o modelo de gerenciamento usando XML e mostra como as ferramentas de domínio público auxiliaram no desenvolvimento de um protótipo do agente XML.*

Abstract. *Programmable controllers became deciding factor to control plant floor process. Allow the management of those elevates them to the same group of another networking equipments. This is possible through XML language, nowadays it has a larger rise and acceptance. This paper describes the features of programmable controllers engaged in the process control, an exhibition of the model of management using XML and showing off the tools of public domain helped in the development of a XML agent prototype.*

1. Introdução

A inserção de controladores programáveis na indústria, data de 1969 [1], ano seguinte a sua especificação por uma divisão da General Motors Corporation [2]. Esses tinham a meta de reduzir a complexidade e o alto custo da utilização, até então, dos painéis de relés. Substitutos dos quadros de relés, os controladores programáveis permitem maior flexibilidade dos elementos e componentes envolvidos no controle dos processos submetidos a esse.

O gerenciamento desses dispositivos tende a facilitar a identificação de falhas, o controle do processo e demais vantagens trazidas por um bom esquema de gerência. Uma maneira de realizar este gerenciamento é através de programas conhecidos como supervisórios. Além desses, uma nova classe de ferramentas de gerenciamento vem sendo disponibilizada, utilizando protocolos como o SNMP e XML. A proposta deste trabalho inclui a elaboração de um modelo de gerenciamento usando XML. A criação um modelo de gerenciamento baseado em tecnologias como o XML e a *web*, permite desde a utilização de um *browser*, até o desenvolvimento de novas ferramentas, para aquisição/requisição de dados em controladores.

As seções seguintes apresentam o estágio atual desta investigação. Na segunda seção está uma breve explanação do uso de controladores e o controle de processos. O modelo proposto é detalhado na terceira seção. A construção do protótipo, utilizando ferramentas de domínio público, é mostrada na quarta seção. Demais considerações situam-se ao final deste.

2. Controle de Processos

Um Controlador Lógico Programável (CLP) é um sistema de controle industrial. Sendo um sistema, esse é composto por *hardware* e *software*. Todo CLP possui ao menos uma unidade de processamento, memória e interfaces de I/O [2]. Além das interfaces de I/O estão inseridas também as interfaces de comunicação, sendo portas Seriais e *Ethernet* as mais utilizadas. Por estarem inseridos normalmente em um ambiente agressivo, os CLPs são concebidos para suportarem altas temperaturas, baixa umidade e elevado níveis de ruído, comuns no chão de

fábrica. Esse dispositivo em conjunto com o sistema executivo, disponibilizam um ambiente propício a execução da lógica de controle. A forma como controlador atua no controle do processo é mostrado na Figura 1.

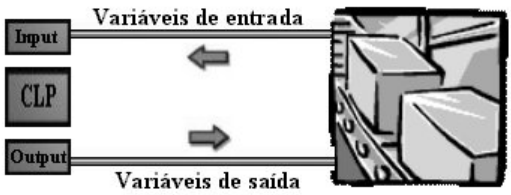


Figura 1. Controle de Processos Típico

O controle da manufatura e máquinas, nas indústrias, realizado anteriormente por painéis de relés é hoje regido por controladores programáveis. Práticos, confiáveis e a custos admissíveis [3] e de fácil adequação a novos controles, esses permitem a indústria, sob a ótica de processos produtivos, capacidade de atender as constantes adaptações exigidas pelo mercado, em um tempo relativamente curto. Além dos aspectos relacionados ao controle do processo, e não menos importante, está a supervisão ou acompanhamento e controle efetivo do processo.

A supervisão de processo é realizada por meio de IHM – *Interface Human Machine* ou aplicações de supervisão executadas em PCs, geradas por softwares que permitem a integração dessas com a lógica de controle. Os supervisórios tais como: *Elipse*, *Fix*, etc. possuem uma série de componentes, além de mecanismos para armazenamento e os *drivers* para o acesso aos controladores de diversos fabricantes. A forma de interação entre a aplicação e a lógica de controle é obtida por meio de associações, conhecidas como *tags*, para tanto são utilizados canais seriais ou TCP/IP sobre *Ethernet*. Modelos alternativos de supervisão estão sendo propostos com o intuito de facilitar a integração com os atuais padrões de mercado. A seção seguinte dedica-se a apresentação de alguns modelos, bem como a explanação do modelo proposto nesta pesquisa.

3. Modelo de Gerenciamento

O acompanhamento em tempo real, bem como a intervenção no processo dá ao gerente o controle efetivo do processo. Permitir que o controle seja realizado não somente por equipamentos conectados ou relativamente próximos ao controlador, portanto no chão de fábrica, vem sendo o principal propósito de estudos na tentativa de encontrar meios alternativos para o gerenciamento, tanto de controladores quanto do processo sob controle.

Em [3] é encontrado uma proposta para o gerenciamento de controladores programáveis utilizando-se o SNMP, tendo essa a vantagem de permitir a integração dos controladores as plataformas de gerência já existentes. Outro exemplo é o *WebGate*, nome dado pela empresa Altus ao seu *gateway* responsável pela comunicação entre equipamentos conectados à redes *Ethernet* e os controladores da própria empresa ligados à redes de campo.

O modelo proposto (Figura 2) neste trabalho considera primordial os aspectos relacionados ao controle do processo, possibilitando o acesso aos operandos envolvidos no controle através de mecanismos simples como o uso de um *browser*, por exemplo. Para que haja o gerenciamento do processo é necessário a troca de dados entre uma aplicação ou um *browser* e o controlador, para tanto será utilizado a linguagem XML e, uma DTD genérica descreve os objetos de interesse em controladores de diversos fabricantes, garantindo efetivamente a consistência dos documentos XML gerados. A DTD e as instâncias de documentos XML são transmitidas a um *parser* de validação [4] para o confronto dessas, resultando em instâncias consistentes. Os dados contidos em elementos e atributos nas instâncias de documento XML

norteiam as requisições feitas ao controlador produzindo, a seguir, respostas à aplicação ou ao *browser* com os dados acerca do processo.

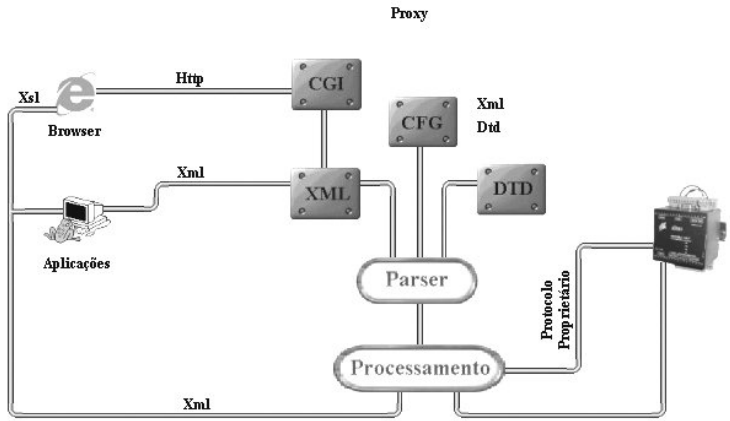


Figura 2. Modelo de gerenciamento de processos em controladores programáveis usando XML, utilizando um proxy.

A implementação do modelo, descrito acima, poderia ser realizada no próprio controlador como no SLC 5/05, da Allen-Bradley, que agrega o SNMP ou em um módulo separado como o *WebGate*, da empresa Altus, sendo essas alterações na maioria das vezes uma tarefa um tanto árdua. Uma solução, com menor grau de dificuldade é a utilização de um *proxy* [5] suportando o modelo exibido na Figura 2 e o uso de software livre na construção do agente XML. Um PC modesto serve ao propósito da implementação aqui sugerida.

O modelo consiste de solicitações advindas de um *browser* convertidas em instâncias de documentos XML ou do envio de documentos XML, diretamente, por outras aplicações. Um *parser* é o responsável pela validação do documento XML contra a DTD, que garante um vocabulário específico e consistente, sendo esse considerado um pré-processamento. Na etapa seguinte, o processamento propriamente dito, são extraídos da instância XML os dados, tais como: endereços, tipos de operandos, bases, etc. os quais serão enviados ao CLP utilizando uma interface serial e o protocolo proprietário para obtenção dos dados, em tempo real, do processo sob controle. Ainda, nesse passo, são produzidas as instâncias de documentos XML com as respostas a solicitação do gerente e enviadas ao *browser*, o qual pode apresentá-las com o auxílio da linguagem XSL ou de aplicações capazes de consumir documentos XML.

4. O Desenvolvimento do Protótipo usando Softwares Abertos

A construção do agente XML passa, antes, pelo desenvolvimento de um protótipo implementado na forma de um *proxy* [5], encarregado de mapear as solicitações XML em requisições proprietárias disponíveis, atualmente, na maioria dos controladores. Um PC AMD K6 II 400 MHz, 96 MB de memória RAM e uma partição de 4 GB no HD vem sendo utilizado como tal. O equipamento dispõe, até agora, do sistema operacional Red Hat Linux *release* 7.1 ***(www.redhat.com)***, do servidor *web* Apache 1.3.19-5 ***(www.apache.org)***, de um compilador ANSI C – gcc 2.96-81 ***(http://gcc.gnu. org)*** e de um conjunto de funções utilizadas na manipulação dos documentos XML. Ainda, são necessárias as recomendações XML, DTD e XSL do W3C ***(www.w3.org)*** referentes ao conteúdo, a criação de um vocabulário específico e à apresentação respectivamente, além da especificação do protocolo de comunicação com o controlador. Todos, a exceção dessa última, são de domínio público.

A entrada, bem como a saída do *proxy* deve, necessariamente, constituir-se de documentos XML. Uma biblioteca com as funções de um *parser* com validação pode ser facilmente localizada em *sites* apropriados: Expat ***(http://expat.sourceforge.net/)***, Xerces ***(www.apache.org)*** e Libxml ***(http://xmlsoft.org)*** são algumas amostras das quais podem ser utilizadas para essa finalidade. Um comparativo entre essas bibliotecas e outras mais, está disponível em ***http://xmlsoft.org***. Por sua plausível documentação e facilidade de adaptação, pela equipe de desenvolvimento, a biblioteca libxml2 2.5.6 [6] foi utilizada na construção do protótipo. Suas funções possibilitam a validação das instâncias XML de requisições e a extração dos dados necessários ao mapeamento das solicitações do gerente ao CLP, em Alnet I [7].

As ferramentas de softwares livres não são as únicas opções para a implementação do protótipo. Podendo, para tanto, utilizar-se do *Microsoft* Windows XP Professional (R$ 1.199,00) em conjunto com o IIE e uma ferramenta de desenvolvimento em linguagem C como o *Microsoft* Visual C++ (R$ 449,00), fonte: ***http://www.brasoftware.com.br***, acesso em abril de 2003. Essas possuem um custo considerável e exigem um equipamento com as configurações superiores, ao mencionado anteriormente, para o desenvolvimento do agente XML.

Como uma aplicação capaz de gerar documentos XML de solicitações, ainda, não está disponível, um módulo CGI foi desenvolvido. Esse foi implementado utilizando o cgic 1.07 [8], combinado as alterações feitas na configuração do servidor *web* (Apache) para execução de programas CGI - *Common Gateway Interface*, para avaliar os dados de um formulário e, assim, obter a solicitação do supervisor, gerando uma instância do documento XML de requisição.

5. Considerações Finais

O PC utilizado no desenvolvimento do protótipo, descrito anteriormente, não necessariamente deve ser colocado na função de *proxy*. Apesar desse ser recomendado, um computador do tipo *Intel* 486, 16 MB de RAM e HD com aproximadamente 500 MB, já seria suficiente para abrigar os principais módulos do Kernel do Linux, o servidor *web* Apache e o código do agente XML, atualmente, com 52 KB. A prova disso é que no futuro a agente deverá ser agregado ao CLP.

De todas as ferramentas utilizadas, na construção do protótipo, somente a especificação do protocolo de comunicação com o CLP, seja ele da Altus ou de outra empresa, não é de domínio público e aberto. No entanto, o modelo de gerenciamento usando XML, proposto aqui, cria uma interface padrão e aberta possibilitando a integração de diversos controladores de diferentes fabricantes.

6. Referências

[1] Miyagi, Paulo E. “Controle Programável - Fundamentos do Controle de Sistemas a Eventos Discretos”. São Paulo: E. Edgard Blücher, 1996.

[2] Jones, C. T. and Bryan, L. A.. “Programmable Controllers – Concepts and Applications”. IPC First Edition, 1983.

[3] Cervieri, Alexandre; Netto, João César; Granville Lisandro Zambenedetti. “An Approach to Manage Programmable Controllers using SNMP and MIBs”. International Conference on Telecommunication Systems (9. : 2001 mar. 15-18 : Dallas). Modeling and Analysis : Proceedings. [S.l. : s.n.], 2001. 638p : il.

[4] Anderson, Richard et al. “Professional XML”. RJ: Ed Ciência Moderna Ltda, 2001.

[5] Stallings, William. “SNMP, SNMPv2, SNMPv3, RMON 1 and 2”. 3. ed. USA: Addison-Wesley, 1999.

[6] Veillard, Daniel. “The XML C library for Gnome – Libxml”. http://xmlsoft.org.

[7] Altus Sistemas de Informática SA. “Norma Técnica do Protocolo Alnet I”.Porto Alegre, 1995 (NTP-031).

[8] Boutell, Thomas. “cgic: An ANSI C library for CGI Programming”. http://www.boutell. com/cgic.

Astrha/E – Ambiente Java/XML que Implementa em Hiper-Animações estruturadas por Máquinas de Mealy

Roges Horacio Grandi, Paulo Fernando Blauth Menezes

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91501-970 – Porto Alegre – RS – Brazil

{roges, blauth}@inf.ufrgs.br

Abstract. *Hyper-Animations are media that where hypertechnology, multimedia and computer animation techniques are built in together. Astrha is a research project where customized Mealy machines are used as hyper-animation frameworks. It is divided in three parts: Astrha/M, theoretical framework model; Astrha/L, a set of four XML dialects (mealy, style, hyper and environment) used to, respectively, to describe Astrha/M machines, text styles, hypermedia files and environments; Astrha/E is a Java environment where Astrha/L codes are interpreted in order to offer the hyper-animated desired semantics. All the implementation is under open source code and is specially recommended to build up simulators and educational media.*

Resumo. *Hiper-Animações são mídias que reúnem teoria hipermídia, multimídia, com técnicas de animação por computador. Astrha é um projeto de pesquisa no qual máquinas de Mealy especializadas são utilizadas para estruturar hiper-animações. Divide-se em três partes: Astrha/M, modelo teórico estrutural; Astrha/L, conjunto de quatro dialetos XML (mealy, style, hyper e environment) utilizados, respectivamente, para descrever máquinas Astrha/M, estilos de texto, arquivos hipermídia e ambientes; Astrha/E, ambiente Java que lê códigos Astrha/L, oferecendo a semântica desejada. Toda implementação é realizada com código aberto, sendo especialmente adequada para a construção de simuladores e programas educacionais.*

1. Introdução

Hiper-animação é um conceito que foi apresentado por (Kappe 1990). Conforme o autor, uma hiper-animação é um conceito para criação de animações interativas, combinando, essencialmente, tecnologias da Computação Gráfica e Hipermídia. Kappe denomina seqüências simples de animação "grupo de documentos" *(document cluster)*, e a visita a um grupo de documentos é definida como um roteiro *(tour)*.

Máquinas de Mealy são autômatos finitos modificados de forma a gerar uma palavra de saída a cada transição (Menezes 2001, p. 79). O projeto acadêmico Astrha propõe-se a validar a utilização de máquinas de Mealy especializadas para estruturar hiper-animações. Para tal, sua concepção foi dividida em três passos incrementais: modelo teórico estrutural (Astrha/M), linguagem (Astrha/L) e ambiente (Astrha/E).

2. Astrha/M (*Model* - Modelo Estrutural)

Hiper-animações são mídias com nível diretivo de interatividade[1]. Dessa maneira, autômatos finitos com saída são especialmente adequados para estruturar, depurar e manter códigos de diálogos humano-computador. Cada estado ou transição ou estado pode ser documentado, tratado ou depurado, se assim especificado.

Astrha/M é um modelo especializado de Máquina de Mealy que possibilita a construção de estruturas para unidades gráficas que, em um meio gráfico e dinâmico representem instâncias de hiper-animações. Trata-se de um modelo não-determinístico e reflexivo, podendo ser representada como uma 6-upla, conforme mostra a Figura 1.

$M = (\Sigma_\tau, Q, q0, \Delta, \Omega, \lambda)$ onde:

a) Σ_τ é um conjunto apontado finito de símbolos de entrada, onde é τ é o elemento distinguido.
b) Q é um conjunto finito de estados.
c) q0 é estado inicial, o qual pertence a Q.
d) Δ é um conjunto finito de símbolos de saída , denotando mídias.
e) Ω é um conjunto de palavras de saída, tal que $\Omega \subset \Delta^* \wedge \varepsilon \in \Omega$.
f) λ é uma função parcial de transição $\lambda : Q \times \Sigma_\tau \rightarrow 2^{(Q \times \Omega)}$, tal que $\forall q \in Q \; \lambda(q, \tau) = \{(q, \varepsilon)\}$

Figura 1 – Máquina de Mealy especializada para estruturar hiper-animações

Note-se que as palavras de saída são predefinidas, reduzindo seu conjunto a uma quantidade de elementos finitos, com o objetivo de definir com clareza quais são as palavras de saída desejadas para o ambiente. É necessário que os símbolos de entrada formem um conjunto apontado (Menezes 2002, p. 109) para definir o elemento distinguido τ que levará uma transição a ser reflexiva, cujo próximo estado é o estado atual e cuja palavra de saída (ε) é vazia. As transições reflexivas têm a semântica de inoperabilidade (*no operation*, ou *nop*), onde as computações realizadas durante a transição não são percebidas externamente.

3. Astrha/L (*Language*- Linguagem)

A forma escolhida para traduzir o modelo teórico estrutural de Astrha/M para Astrha/E foi descrever uma linguagem de marcação XML, composta de quatro dialetos, denominada Astrha/L.

O dialeto *mealy* é utilizado para traduzir Astrha/M através de uma linguagem de programação. Sendo as hiper-animações mídias compostas, uma palavra de saída descrita no dialeto *mealy* representa uma seqüência animada complexa na definição de (Kappe 1990). Os demais dialetos de Astrha/L, *hyper*, *style* e *environment* descrevem,

[1] "Multimídia com nível diretivo de interatividade permite que o usuário inicie e responda a ações internas da aplicação, assim como adaptar aspectos do ambiente como escolha de cor, tipo de retorno, etc. (Heller 2001)."

respectivamente, hiperligações, estilos e ambiente. Esses três dialetos têm influência direta da tecnologia do *World Wide Web Consortium* (W3C). *Hyper* foi influenciado especialmente por XLink, no que se refere aos conceitos de ligações estendidas. Texto sublinhado, mapas clicáveis e em cor diferenciada como padrão de escrita para âncoras de um hipertexto é oriundo do HTML. *Style* é uma tradução em XML de um subconjunto de atributos HTML/CSS para textos e hiperligações. O dialeto *environment* teve como base tecnológica de inspiração SMIL Animation (W3C 2001). A semântica de uma animação com suas hiperligações em SMIL Animation corresponde, parcialmente, a de uma hiper-animação em Astrha/L. Podemos perceber uma forte aproximação e influência da linguagem Astrha/L com a tecnologia e conceitos do W3C.

4. Astrha/E (*Environment* – Ambiente)

Astrha/E implementa um ambiente dinâmico, gráfico e interativo para Internet, que lê e interpreta definições de hiper-animações baseadas em máquinas de Mealy escritas em Astrha/L. Ao ser executado, Astrha/E permite que interações com o usuário através de hiperligações definam, dinamicamente, a seqüência de mídias a serem apresentadas.

Modelado em UML, foi construído em linguagem Java e utiliza software livre e gratuito em todos os seus componentes, facilitando sua distribuição e favorecendo a produção de mídias para educação a distância. A Figura 2 mostra o pacote de mídias de Astrha/E em nível conceitual. Nesse pacote, podemos perceber que uma hiper-animação é composta por hiperligações *(links)* e variados tipos de mídia: sons *(sounds)*, figuras *(figures)*, textos *(hypermedium* e *paragraph)*. Permite, inclusive, composição com outras hiper-animações.

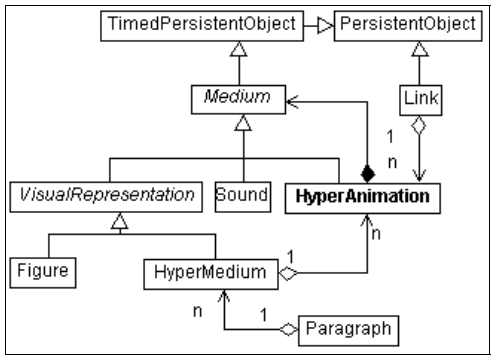


Figura 2 – Pacote de mídias de Astrha/E em nível conceitual

O diagrama de implantação da Figura 3 mostra os componentes utilizados, todos de tecnologia aberta. Java Applets[2] e JAXP[3], da *Sun Microsystems*; Crimson[4], da Apache e JDOM[5] da organização do mesmo nome.

[2] Programas escritos em linguagem Java que podem ser incluídos em uma página HTML.

[3] Biblioteca Java para processamento de arquivos XML.

[4] Analisador XML suportado pela biblioteca JAXP.

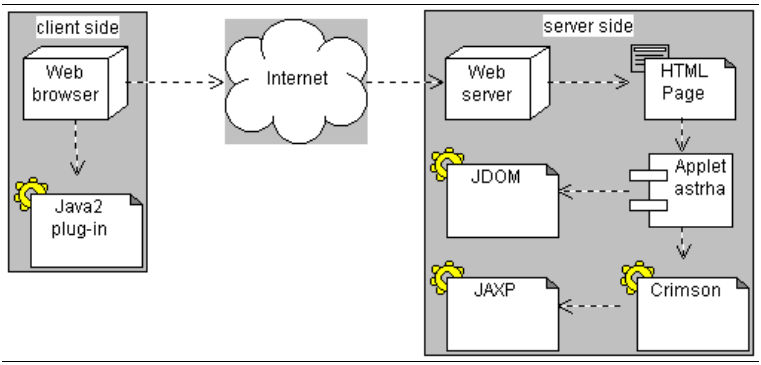


Figura 3 – Visão geral de um ambiente cliente-servidor para Astrha/E

5. Conclusões

Em experimentos realizados em laboratório, pudemos comprovar que tecnologia aberta tem a capacidade de implementar, totalmente e com vantagens, hiper-animações estruturadas por máquinas de Mealy especializadas, que é o objetivo do projeto de pesquisa Astrha, sendo a combinação das tecnologias Java/XML uma escolha bem sucedida para garantir portabilidade dentro do paradigmas da orientação a objeto para sistemas Internet. Para conhecer mais sobre o projeto Astrha: http://teia.inf.ufrgs.br.

6. Agradecimentos

Este trabalho é parcialmente suportado pelo CNPq (projetos Hyper Seed, HoVer-CAM, GRAPHIT), CNPq/NSF (Projeto MEFIA) e FAPERGS (Projeto QaP-For). Agradecimentos extensivos a todos os colegas de pesquisa.

Bibliografia

Kappe, F. M.; Maurer, H. “Animation in Hyper-G – An Outline” In Haase V. and Zinterhof P. (editors), *Proc. Future Trends in Information Technology ’90, Salzburg, Austria*, pages 235-248, Austrian Computer Society. Vienna, Munich, Sep. 1990.

Menezes, P. F. B. “Linguagens Formais e Autômatos”**.** 4.ed. Porto Alegre: Sagra Luzzato, 2001.

Menezes, P. F. B.; Hausler, Edward Hermann. Teoria das Categorias para Ciência da Computação. 1. Ed. Porto Alegre: Sagra Luzzato – UFRGS, 2002. 324 pp.

Heller, R. S. et al. “Using a Theoretical Multimedia Taxonomy Framework.” ACM Journal of Education Resources in Computing, v.1, n.1, 2001.

W3C World Wide Web Consortium. “SMIL Animation - W3C Recommendation 04-September-2001”, http://www.w3.org/TR/2001/REC-smil-animation-20010904 Jan. 2003.

[5] Biblioteca Java que facilita a leitura, manipulação e escrita de arquivos XML.

A Free Software for the Development of Adaptive Automata

Hemerson Pistori[1 2] **, João José Neto**[2]

[1]Depto. Engenharia de Computação – Universidade Católica Dom Bosco
79117-900 Campo Grande, MS, Brasil

[2]Laboratório de Linguagens e Tecnologias Adaptativas
Escola Politécnica – Universidade de São Paulo
05508-900 São Paulo, SP, Brasil

pistori@ec.ucdb.br, joao.jose@poli.usp.br

Abstract. *Although recent, adaptive technology is already being used to solve complex problems in areas such as compiler construction, natural language processing, computational vision and robotics. This papers introduces AdapTools, a free-software environment that helps the development of solutions based on adaptive automata. Since adaptive automata generalize finite state and structured pushdown automata, AdapTools may also be used as a tool for the development of traditional automata, as well as an educational tool.*

1. Introduction

The former traces of adaptive devices, which achieved its most general formalization in [Neto, 2001], appeared in the early work of van Wijngaarden [van Wijngaarden, 1974]. Although van Wijngaarden's two-level grammars cannot be considered adaptive devices, in the sense that they do not change its own structure at execution time, all of them increase the representative power of formal devices, like context-free grammars, without loosing much of its "ease of use".

The last thirty years witnessed the upcoming of several types of adaptive devices, all of them adding some self-modifying or dynamic characteristic to a specific formalism, in order to overcome its limited expressive power. Among them are: modifiable grammars [Burshteyn, 1992]; self-modifying finite automata [Rubinstein and Shutt, 1994] and dynamic grammars [Boullier, 1994]. An attempt to generalize the above concepts, capturing the meaning of adaptiveness and divorcing it from the concept of a subjacent device (e.g. context free grammars, finite-state automata) gave rise to the formalism called rule-driven adaptive device. This work paved the way for the construction of adaptive devices based on subjacent mechanism as diverse as Markov chains [Basseto and Neto, 1999] and decision trees [Pistori et al., 2003]. Examples of applications of adaptive devices may be found in a variety of areas, like automatic music composition [Basseto and Neto, 1999], natural language processing, robotic navigation [Junior et al., 2000], pattern recognition [Costa et al., 2002] and computational vision [Pistori et al., 2003].

Though most of the adaptive theory is now mature, it still lacks some widely accepted related software tool. Most works on implementation of adaptive devices developed so far are deeply attached to specific projects or to specific subjacent mechanisms, and may not be easily adapted for use in other situations. The present work is a significant attempt in this direction, represented by the software named AdapTools, described below.

Adaptools is still far from implementing the full generality of the adaptive device theory. Nonetheless, its design foresees many of such extensions. Currently, AdapTools implements a virtual machine that executes a slightly modified version of the originally described adaptive automata [Neto, 1993, Neto, 2000] enhanced with an integrated development environment containing some graphical visualization, debugging, project maintenance and editing resources.

Adaptools is distributed under GNU's Public License (GPL), being available at the laboratory of adaptive languages and technologies' web site[1].

Besides having been designed as a development environment for adaptive devices, Adaptools may also be used as an excellent educational tool for disciplines like computer theory and compiler construction. Since the traditional finite state automata, non-deterministic ϵ-automata and structured pushdown automata are all specializations of adaptive automata, AdapTools may be used as a virtual laboratory where the student may practice experimentally the concepts. Several illustrating examples have been included in the package. One of the most important is a compiler-compiler that converts a grammar specification, in Wirth notation, of a language L, and produces an adaptive automaton that recognizes L (this automaton may be simulated by AdapTools); an extremely simple and powerful prototype of a text-to-speech system and an error-recovery scheme for handling simple syntactical errors are also available. Adaptools is already being used as an extra-class educational reinforcement tool in some disciplines of the computer engineering program offered in our institution.

The following section presents a brief description and an example of adaptive automata, as defined in Adaptools. Next, in section 3, an overview of AdapTools features is shown. The last section is reserved for conclusions and future work.

2. Adaptive Automata - Adaptools Version

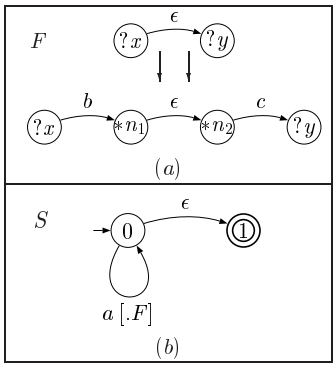
Conceptually, an structured pushdown automaton may be regarded as a set of finite-state automata (called sub-machines) that allows two additional kinds of transitions: sub-machine calls and returns, specifying changes in the flow of execution from one sub-machine to another, with a pushdown stack for holding return addresses. An adaptive automaton is an structured pushdown automaton that allows dynamic modifications in its set of transitions. Such changes are accomplished by performing the so-called query, deletion and insertion elementary actions, grouped in adaptive functions, that are called from and may operate on any transition, before or after its execution. See [Neto, 1993] for more details.

In Adaptools, the form chosen to represent adaptive automata is a table with seven columns and four basic kinds of lines: normal transition, sub-machine calls and returns, and query, deletion an insertion elementary adaptive actions. A *header* column holds names of sub-machines and adaptive functions. In lines representing elementary adaptive actions, the header also informs the type of action: $?$,$-$ or $+$, for query, deletion or insertion, respectively. The columns for *origin and destination states* and *input and output symbols* have the usual interpretation, except that, for adaptive functions, they may refer to variables (in order to express patterns), and generators, representing states that must be created dynamically. Variables and generators are denoted with prefixes $?$ and $*$, respectively. The $push$ column is used in sub-machine calls, to indicate the return state. Sub-machine returns are indicated by the reserved keyword pop placed in the *destination state* column. The last column, for *adaptive action*, is used to link a transition to a pair of adaptive functions and contains a string with the form $B.A$, where B and A are names of adaptive functions that are to be executed Before and After the transition (any one of them may be omitted). The reserved keyword nop may be used anywhere to indicate that the column is not being used. Empty transitions hold the string eps instead of an input symbol.

In order to avoid extra structures, the initial and final states of an automaton are encoded in a somewhat cumbersome way: The $push$ column must have the keyword fin when the *destination state* is a final state, and, by convention, the *origin state* of the first line of each sub-machine always denotes its initial state.

Figure 1 shows an adaptive automaton that recognizes the context-dependent language $a^n b^n c^n$, using both Adaptools table representation, figure 1.(c), and a graphical representation for its only adaptive function, figure 1.(a), and sub-machine, figure 1.(b). The device works as follows: each time some symbol a is consumed (line 6) the adaptive function F finds the only empty

[1] http://www.pcs.usp.br/~lta - download section



	Head	Orig	Inpu	Dest	Push	Outp	Adap
1	$?F$	$?x$	eps	$?y$	$?z$	nop	nop
2	$-F$	$?x$	eps	$?y$	$?z$	nop	nop
3	$+F$	$?x$	b	$*n1$	nop	nop	nop
4	$+F$	$*n1$	eps	$*n2$	nop	nop	nop
5	$+F$	$*n2$	c	$?y$	$?z$	nop	nop
6	S	0	a	0	nop	nop	$.F$
7	S	0	eps	1	fin	nop	nop

(c)

Figure 1: Adaptive Automaton that Recognizes $a^n b^n c^n$
(a) Adaptive Mechanism (b) Subjacent Mechanism (c) Adaptools Object Code

transition of the automata (line 1), removes it (line 2) and insert two new transitions consuming the string bc (lines 3 and 5) with and empty transition strategically put between them (line 4).

In the original definition of adaptive automata [Neto, 1993], variables and generators are explicitly declared and adaptive function calls are allowed to appear inside adaptive functions. Avoiding such features in AdapTools originates a simpler tabular representation of the machine. Implicit declaration of variables and generators certainly do not affect the expressiveness of the device, but an study of the impact of not permitting adaptive function outside the subjacent layer is yet to be conducted. However, the broad range of cases already implemented in AdapTools, including regular, context-free and context-dependent languages, suggests that this modification preserved much of the power of the original device, which is Turing-powerful.

3. Adaptools

Adaptools is a software environment where adaptive automata can be implemented, tested and experimented. The core of this software is a virtual machine that executes adaptive automata represented in tabular form, such as the one depicted in figure 1.(c). The execution of transitions and elementary adaptive actions, as well as the contents of stacks, variables and generators, may be traced, step by step. A powerful graph drawing tool, based on the ForceDirect algorithm featured by OpenJGraph package, may be used for the visualization of modifications accomplished by the automata. The algorithm models nodes and edges as electrical forces and logarithmic springs, respectively, and uses the physical laws of repulsion and tension to automatically find a balanced layout for the graph representing the automaton. By allowing human intervention, pretty good animations are obtained, that really enhance the visualization and comprehension of the devices being simulated. Another important feature is the simultaneous execution of multiple devices, by using Java multi-threading resources. An example of this feature, available in the package, is a parser and a lexical analyzer, that work simultaneously to produce object code for Adaptools' virtual machine. An enhanced version of this example, which features syntactical error-recovering capabilities in the generated code is another powerful tool easily generated within AdapTools, used in compiler construction courses.

4. Conclusions

In this work we introduced a free-software which can be used in, at least, three ways: (1) in the development of solutions using the powerful emerging technology based on adaptive automata, (2) as an educational tool for teaching and training in adaptive automata[2], and weaker traditional

[2] A somewhat related case study can be found in [Leonardi, 1999]

formalisms incorporated by them, like finite-state automata and structured pushdown automata and (3) as the basis for the development of a more general tool that will handle adaptive devices with subjacent mechanisms other than automata. The development of a generalized version of AdapTools that works with adaptive decision trees is already in course.

References

Basseto, B. A. and Neto, J. J. (1999). A stochastic musical composer based on adaptative algorithms. In *Anais do XIX Congresso Nacional da Sociedade Brasileira de Computação. SBC-99.*, volume 3, pages 105–130, PUC-RIO, Rio de Janeiro, Brazil.

Boullier, P. (1994). Dynamic grammars and semantic analysis. Technical Report 2322, INRIA.

Burshteyn, B. (1992). USSA - universal syntax and semantics analyser. *ACM SIGPLAN Notices*, 27(1):42–60.

Costa, E. R., Hirakawa, A. R., and Neto, J. J. (2002). An adaptive alternative for syntactic pattern recognition. In *Proceeding of 3rd International Symposium on Robotics and Automation, ISRA*, pages 409–413, Toluca, Mexico.

Junior, J. R. A., Neto, J. J., and Hirakawa, A. R. (2000). Adaptive automata for independent autonomous nagivation in unknown enviroment. In *Proceedings of IASTED International Conference on Applied Simulation and Modelling - ASM 2000*, Banff, Alberta.

Neto, J. J., Pariente, C. B. and Leonardi, F. (1999). Compiler construction - a pedagogical approach. In *Proceedings of the V International Congress on Informatic Engineering - ICIE 99*, Buenos Aires, Argentina.

Neto, J. J. (1993). Contribuição à metodologia de construção de compiladores. *Tese de Livre Docência, Escola Politécnica, Universidade de São Paulo*. [In Portuguese]

Neto, J. J. (2000). Solving complex problems efficiently with adaptative automata. In *Conference on the Implementation and Application of Automata - CIAA 2000*, Ontario, Canada.

Neto, J. J. (2001). Adaptative rule-driven devices - general formulation anda case study. In *CIAA'2001 Sixth International Conference on Implementation and Application of Automata*, pages 234–250, Pretoria, South Africa.

Pistori, H., Neto, J. J., and Costa, E. R. (2003). Utilização de tecnologia adaptativa na detecção da direção do olhar. In *Anais da Conferencia Internacional de la Sociedad Peruana de la Computacion SPC'2003*, Lima, Peru. [In Portuguese]

Rubinstein, R. and Shutt, J. N. (1994). Self-modifying finite automata. In *Proceeding of the 13th IFIP World Computer Congress*, pages 493–498, Amsterdam: North-Holland.

van Wijngaarden, A. (1974). Revised report on the algorithmic language algol 68. *Acta Informatica*, 5:1–236.

Aplicações SCADA utilizando GNU/Linux

Felipe Damasio, Guilherme Kunz, Marco Andrei Kichalowsky

Elipse Software
Av. 24 de Outubro, 353, 10o. andar – 90510-002 – Porto Alegre – RS – Brasil
{felipewd, guilherme, mandrei}@elipse.com.br

Abstract*. This paper discusses the use of GNU/Linux in the automation and control environments, specially in SCADA applications, showing examples and advantages of this operational system.*

Resumo*. Este artigo trata da utilização do sistema operacional GNU/Linux em ambientes de automação e controle, em especial para aplicações SCADA, apontando exemplos e vantagens desse sistema.*

1 Introdução

O sistema operacional GNU/Linux vem ganhando a atenção de vários setores do mercado corporativo mundial devido a sua estabilidade, escalabilidade e principalmente seu baixo custo.

O apoio de grandes empresas de informática como IBM, Oracle e Intel ao sistema operacional "do pingüim" tem sido um dos grandes motivos para a migração de empresas para a plataforma, buscando desenvolver soluções de baixo custo, mas de alta estabilidade e desempenho. Um exemplo disso é o uso cada vez maior em *embedded systems* (PDAs, televisores, relógios digitais etc.).

Recentemente, diversos fabricantes de sistemas SCADA (*Supervisory Control and Data Acquisition*) começaram a perceber as vantagens do GNU/Linux como base para seus produtos, em conjunto com outras tecnologias como *clusters*.

Esse artigo mostra algumas aplicações deste sistema operacional no ambiente de automação e controle. O documento é dividido da seguinte forma: na seção 2 é introduzido o conceito de *embedded systems*; na seção 3, o conceito de *clusters*; na seção 4 é comentada a aplicação destes conceitos em servidores de aplicações.

2 Embedded Systems

O termo *embedded system* designa uma combinação de hardware e software que desempenha uma função específica. Em alguns casos, embedded systems são partes de um sistema ou produto maior.

Um exemplo interessante dessa possibilidade é o projeto DigiAlbum do grupo de pesquisas de Engenharia da Computação da Universidade Hanyang, Coréia do Sul. O projeto criou um aparelho para Internet que permite ver, salvar e recuperar imagens e vídeo, além de navegar na Web com eficiência. Nesse projeto foi utilizado o Linux como um "esqueleto" para um kernel que cumpre as funções de gerenciamento de memória, processamento de vídeo e controle de E/S. O resultado foi um sistema operacional que ocupa ao todo, 800KB da memória Flash ROM do aparelho. O micro-kernel é responsável por apenas 300KB [Lee, 2001].

Este fato do código fonte do GNU/Linux estar disponível para quem quiser ver, modificar e até mesmo comercializar versões modificadas, foi o que motivou empresas como Sony, IBM, Motorola, entre várias outras a investir pesado no porte do sistema operacional para

várias arquiteturas. Atualmente o GNU/Linux é suportado em 17 arquiteturas, sendo uma delas a arquitetura ARM, que é baseada em processadores como o RISC StrongARM e XScale da Intel, amplamente utilizados em PDAs como o Compaq iPAQ e o Yopy. [Guy, 2003]

Algumas empresas se especilizaram em desenvolver kits de desenvolvimento voltados a embedded systems, contendo compiladores, depuradores e várias outras ferramentas otimizadas para o desenvolvimento de aplicativos embedded, como o Embedded Linux da MontaVista, que será utilizado no celular da Motorola A760. [Montavista, 2001]

O uso de embedded systems no ambiente de automação e controle é tradicional, mas a utilização do GNU/Linux como base para estes sistemas é recente. Como já introduzimos anteriormente, uma das fortes características do sistema GNU/Linux é possuir código fonte aberto e de livre distribuição. Isto possibilita, principalmente, o desenvolvimento de sistemas para automação industrial que utilizam o máximo da capacidade de seu hardware, pois temos a possibilidade de dimensionar e otimizar o sistema operacional para a aplicação que será utilizada. Além disso, é um sistema que suporta um grande número de arquiteturas de hardware, rodando desde supercomputadores até PDAs oferecendo liberdade para a escolha da plataforma de modo que atenda os requisitos do projeto, levando em conta aspectos como: consumo de energia, desempenho e custo. [Kastner et alli, 1999]

Um exemplo nacional da utilização do GNU/Linux em embedded systems é a interface homem-máquina IHM1260T/ElipseSCADA. A interface é baseada em uma versão otimizada do GNU/Linux para rodar na CPU Crusoe 5400/500Mhz da Transmeta. O sistema foi personalizado para utilizar a tecnologia LongRun da Transmeta, que diminui o clock do processador quando submetido a altas temperaturas. [Advantech, 2003]

Complementando o equipamento há o software Elipse SCADA, que serve de base para o desenvolvimento de aplicações para automação e controle industrial. Nesse caso específico, o software permite a ligação de controladores lógicos programáveis a elementos gráficos da interface, possibilitando uma fácil compreensão e manipulação dos dados, além do gerência e controle total do processo automatizado. [Elipse, 2003]

3 Clusters

Cluster é um conjunto de computadores trabalhando para executar determinados tarefas. A justificativa do uso dessa técnica é o fato de que na maioria do tempo, os computadores não utilizam 100% dos seus recursos. Utilizando um cluster é possível ter o máximo proveito dos equipamentos.

Com o aumento da utilização da arquitetura *thin-client* na automação, controle e supervisão de processos, temos a utilização de um cluster na substituição de um servidor central de aplicações, visando diminuição de custos e escalabilidade.

A unidade básica de um cluster é um computador, também chamado de “nodo”. Através da adição de mais máquinas, o cluster se torna mais escalável, ou seja, mais processos poderão ser distribuídos entre os diferentes nodos. Existem basicamente três classes de clusters: *Fail-over*, HPC (*High Performance Computing* ou Computação de Alta Performance) e *Load-Balancing*.

Clusters *Fail-over* consistem de dois ou mais nodos conectados utilizando uma conexão *heartbeat*. A conexão *heartbeat* é utilizada para monitorar todos os serviços oferecidos pelo nodo mestre constantemente, para que no caso de algum deles não estar disponível, um outro nodo do cluster assumir o serviço, mantendo o servidor sempre disponível.

Clusters do tipo HPC são utilizados para cálculos computacionais extremamente complexos, onde um processo é dividido entre vários nodos do cluster para ser computado num

tempo aceitável. Como exemplo, temos a utilização da técnica para resolução de cálculos científicos avançados, mas também é muito utilizada para a quebra de chave RSA (criptografia) e SETI (Search for Extra Terrestrial Intelligence).

Uma implementação clássica de clusters do tipo HPC é o projeto Beowulf. Este projeto começou no início de 1994, numa sub-divisão de pesquisa da NASA, coordenado por Donald Becker e Thomas Sterling. Na época havia a necessidade de uma arquitetura de baixo custo para processamento não-linear (cálculos como previsão de tempo) como alternativa aos supercomputadores, que eram extremamente caros.

A última classe de clusters são os do tipo *Load-Balancing* ou balanceamento de carga. Neste tipo, quando é feita uma requisição (por exemplo, em um servidor Web), o cluster verifica qual nodo é o menos utilizado e repassa a requisição para ele, o que gera uma alta perfomance no serviço como um todo (sendo transparente para o cliente que faz a requisição).

Existem implementações que agregam mais de um tipo de *cluster*. Os *clusters* OpenMosix são do tipo *Load-Balancing* e são utilizados para suprir a necessidade de distribuição e migração de processos originalmente não-distribuídos entre os nodos, característica dos clusters Beowulf [Bar, 2003].

O projeto OpenMosix foi criado por Moshe Bar para ser uma implementação software livre baseada em GNU/Linux do projeto Mosix, coordenado por Amnon Bark. Esse projeto propõe uma solução para o problema dos clusters HPC de necessitar que as aplicações a serem computadas utilizem algoritmos distribuídos. No caso do OpenMosix, não há essa necessidade porque os algoritmos responsáveis pela migração e *Load-Balancing* do cluster estão no kernel do sistema operacional, o que faz com que os processos sejam transparentes para qualquer aplicação utilizada no sistema.

Uma das limitações de clusters OpenMosix é a impossibilidade de migrar processos baseados em *pthreads* (POSIX Threads), sendo esta uma limitação do kernel Linux, o qual não cria LWP (*Light-Weight Process*) com próprio espaço de endereçamento para cada *thread*. Uma possível solução para este problema será a incorporação de DSM (*Distributed Shared Memory*) nos clusters OpenMosix, porém esta característica ainda não está disponível no projeto.

4 Servidores de Aplicação

Um exemplo da aplicação de clusters em aplicações SCADA é a utilização de servidores de aplicativos, onde pode-se manter um número *x* de computadores para realizarem o papel de servidores de aplicações Elipse SCADA. Quando um dos clientes solicita iniciar uma nova instância do Elipse SCADA, o nodo mestre analisa o pedido e direciona para a máquina servidora de aplicação que está sendo menos utilizada.

Caso ocorra no projeto uma necessidade de expansão do número de clientes ao ponto de interferir na qualidade dos processos nos clientes, basta adicionar computadores para realizar o papel de servidores de aplicação, sempre mantendo um alto desempenho nos clientes. Isto também auxilia a minimização de custos nos clientes, visto que todo processamento será exclusivamente realizado nos servidores de aplicação e somente os gráficos da aplicação serão retornados [Salvador, 2002].

Finalmente, chega-se a uma arquitetura em que também podemos utilizar o próprio conceito de embedded systems para as máquinas-cliente, tendo uma série de thin-clients de baixo custo utilizando um cluster servidor de aplicações [Salvador, 2002].

5 Conclusão

Existe uma crescente necessidade da diminuição de custos em grandes aplicações de automação industrial. Através da utilização de GNU/Linux é possível construir aplicações de baixo custo obtendo bom desempenho e escalabilidade.

Além disso, a possibilidade de independência tecnológica foi o que atraiu empresas como IBM, Intel e Oracle, que atualmente investem em pesquisas para a otimização e difusão do sistema.

Devido ao investimento dessas e outras empresas, o GNU/Linux está disponível em diferentes classes de aplicações (como *embedded systems* e *clusters*), o que permite que a área de automação e controle possa usufruir desta adaptabilidade para seus diferentes cenários.

Referências Bibliográficas

Advantech Brasil. (2003) “Advantech/Elipse - IHM1260T/Elipse SCADA”. http://www.advantech.com.br/Lancamento.pdf, Abril.

Bar, M. (2003) “OpenMosix vs. Beowulf: a case study”. http://www.democritos.it/activities/IT-MC/openMosix_vs_Beowulf.pdf, Abril.

Elipse Software. (2003) “Parceria Elipse/Advantech”. http://www.elipse.com.br, Abril.

Guy, A. et alli. (2003) “The Familiar Project”, http://familiar.handhelds.org. Abril.

Kastner, W. et alli. (1999) “Linux in factory automation? Internet controlling of fieldbus systems!”. ETFA '99 - 7th IEEE International Conference on Emerging Technologies and Factory Automation, Procedings, vol. 1, pág. 27–31.

Lee, Sang-Yeob et alli. (2001) “Designing low cost embedded system for multimedia data processing: Linux based micro kernel approach”, ICCE - International Conference on Consumer Electronics, Procedings, pág. 346–347.

Montavista Software, Inc. (2001) “Leveraging Linux® for Embedded Applications”, http://www.mvista.com/dswp/Leveraging_Linux.pdf.

Salvador, M. e Correa, A. (2002) “Distribuição e Redundância de Servidores Aplicadas À Melhorias no Conceito de Thin-Client HMI”, Palestra apresentada no Congresso do ISA Show South America 2002.

GoboHide: Uma Solução Flexível e Escalável para Inodes Ocultos no Kernel Linux

Felipe W Damasio **Lucas Correia Villa Real**
felipewd@terra.com.br **lucasvr@gobolinux.org**

Abstract. *This paper presents a robust and flexible solution to allow proposals of restructure of the legacy directory tree of the GNU/Linux operating system to keep full compatibility with the legacy tree, keeping it invisible to the user. A tool for managing the legacy tree was developed and is being used by the GoboLinux distribution.*

Resumo. *Este artigo apresenta uma solução robusta e flexível para permitir que propostas de reestrutração da árvore de diretórios do sistema operacional GNU/Linux mantenham compatibilidade com a árvore de diretórios tradicional sem que esta seja visível para o usuário. Uma ferramenta para administração da árvore clássica foi desenvolvida e está sendo utilizada pela distribuição GoboLinux.*

1. Introdução

Em virtude do ritmo rápido de desenvolvimento do software livre, o processo de instalação e remoção de programas tornou-se comum e freqüente, diferentemente do que ocorria quando do estabelecimento da hierarquia de diretórios do UNIX, na qual as distribuições do GNU/Linux usualmente se baseiam. Os critérios de organização da árvore de diretórios tradicional não levam em conta estas necessidades; torna-se interessante realizar uma revisão destes critérios e buscar uma alternativa [Muhammad and Detsch, 2002].

A adoção de uma nova árvore de diretórios para o sistema operacional GNU/Linux facilita a manutenção e a atualização de programas, bem como a organização geral do sistema. Porém, existe a necessidade de manter a compatibilidade com a árvore de diretórios tradicional do UNIX, pois muitos programas tem *paths hardcoded* baseados na árvore clássica. É interessante um mecanismo que permita que seja mantida a compatibilidade com a árvore clássica de maneira transparente para qualquer processo, permitindo também ao usuário apenas a visualização da nova árvore de diretórios.

Este artigo apresenta uma solução simples, escalável e de fácil administração para dissimular a existência desta árvore de diretórios tradicional, possibilitando que uma nova árvore de diretórios seja usada sem que seja perdida a compatibilidade com aplicações que utilizam a árvore clássica do GNU/Linux.

O restante deste artigo está organizado da seguinte maneira: Na Seção 2 será vista a abordagem utilizada em outros sistemas operacionais, bem como a abordagem utilizada pelo GoboLinux; na Seção 3 será descrita a implementação do GoboHide no kernel Linux; na Seção 4 será descrita uma ferramenta administrativa feita em espaço de usuário que é utilizada pelo administrador do sistema para gerenciar a árvore *legacy* de diretórios; na Seção 5 o artigo é concluído, deixando alguns pontos abertos que podem ser abordados como trabalhos futuros.

2. Abordagens alternativas

Atualmente, a distribuição GoboLinux[1] utiliza a mesma estratégia do sistema operacional Mac OS X[Apple Computer Inc., 2001], no qual *links* são usados para poder ser criada uma nova árvore de diretórios e manter a compatibilidade com programas com *legacy paths hardcoded*.

A grande vantagem deste tipo de abordagem está em sua simplicidade, já que o kernel transparentemente re-envia requisições dos *links* simbólicos aos novos diretórios, mantendo total compatibilidade.

No entanto, como a reestruturação da árvore de diretórios visa simplificar a representação desta para usuários domésticos, o uso de *links* simbólicos traz uma grande desvantagem, já que as duas árvores (a árvore proposta e a clássica) são vistas pelo usuário. Para isso, o Mac OS X utiliza uma estratégia incomum para "esconder" a árvore clássica do usuário. Na interface gráfica é exibida uma árvore de diretórios Macintosh, contendo diretórios como */System/Library* e */Network/Remote_Workstation*. Além disso, a interface exibe alguns diretórios em localizações diferentes da real. Por exemplo, o diretório */Mac OS X* é um *link* para o diretório raiz, e alguns diretórios, como */Applications*, aparecem na interface apenas como */Mac OS X/Applications*. No entanto, acessando-se o sistema de arquivos através de um *shell*, tornam-se acessíveis diretórios UNIX como */usr* e */etc*.

Esta abordagem é somente possível em um ambiente proprietário, onde toda a interface primária do sistema é desenvolvida por apenas uma empresa. Além disso, falhas no *design* como a possibilidade de ver a árvore de diretórios através do uso de um *shell* mostram que a abordagem utilizada pelo Mac OS X é inutilizável em um sistema operacional tão heterogêneo como o GNU/Linux. Na Seção 3 é apresentada a solução adotada pelo GoboLinux para esconder a árvore de diretórios *legacy*.

Nota-se também que o ideal é permitir que seja usada qualquer interface com o usuário e qualquer programa de gerenciamento de diretórios de maneira transparente, acessando (e visualizando) apenas a nova árvore de diretórios sem que seja necessária a modificação desses programas para adaptação à nova árvore de diretórios.

3. Implementação no Kernel Linux

A representação de arquivos no kernel Linux é feita com o uso de identificadores denominados *inode numbers*. Estes números são únicos para todos os arquivos em uma dada partição.

Para realizar a leitura de um diretório, é utilizada a chamada de sistema `readdir`. Esta chamada de sistema realiza a leitura de um diretório especificado, copiando seu conteúdo para uma região alocada na memória. Desta forma, para a implementação do GoboHide, basta verificar para cada elemento lido nesta operação se ele é um dos diretórios ou um link simbólico da árvore clássica. Esta informação é obtida a partir do *inode number* referenciado pelas estruturas de entrada de diretório. Caso o *inode* lido seja um dos que deseja-se ocultar, esta entrada simplesmente não é copiada para a região de memória (operação comumente referenciada como *filldir*), e o próximo inode é processado. Como a chamada de sistema `readdir` é implementada em praticamente todos os sistemas de arquivos, ela torna-se um local plausível para realizar a interceptação.

Através da modificação de apenas esta chamada de sistema, resolvemos o problema de visualização da árvore clássica de maneira transparente para qualquer processo, desde que o sistema de arquivos no qual a árvore clássica está localizada faça a checagem das estruturas GoboHide – composta por itens descritivos sobre os inodes sendo ocultados, disponibilizados em uma lista.

[1] ver `http://www.gobolinux.org`

Além disso, como apenas a chamada de sistema `readdir` está sendo interceptada, *paths hardcoded* em programas que referenciam a árvore clássica funcionarão normalmente, já que as chamadas de sistema `open` e `close` não são modificadas, e pela utilização de links simbólicos o acesso e redirecionamento à nova árvore de diretório é também feito pelo Kernel (na camada VFS) de maneira transparente.

O problema de ser interceptada a chamada `readdir` para consultas na árvore clássica é manter a consistência da lista e permitir que essa estrutura possua um acesso rápido, já que a chamada de sistema `readdir` é utilizada sempre que um processo deseja ver o conteúdo de um diretório.

Para manter a consistência da lista, foram modificadas as chamadas de sistemas `unlink` e `rmdir` na camada VFS, permitindo uma transparência na gerência dos inodes tanto para qualquer processo como para qualquer sistema de arquivos. Além disso, na chamada de sistema `unlink` apenas links simbólicos são checados, já que esta chamada é utilizada para a remoção de *qualquer* arquivo. Com isso, é diminuído o *overhead* de checagens na lista GoboHide, pois apenas remoções de diretórios/links simbólicos fazem consultas a ela.

Métodos de leitura da lista de inodes são feitos de maneira concorrente, porém apenas um metódo que deseja modificar a lista pode ser utilizado por vez (através da utilização de *Read-Write Locks*), sendo que são utilizados algoritmos de complexidade **O(1)** para o acesso a lista de inodes, garantindo um tempo constante de acesso à lista de inodes que deverão ser ocultados, ou seja, independente do número de inodes armazenados na lista.

Também foram utilizados *locks* que salvam o contexto de IRQ e desabilitam interrupções no processador local, não permitindo que a estrutura de checagem de *inodes* seja danificada por acessos concorrentes em um sistema SMP (*Symmetric Multi-Processing*) ou mesmo num sistema UP (*Uni-Processing*), bem como não são permitidos que hajam *deadlocks* na aquisição de *write locks*.

Na implementação do GoboHide, foi utilizado o conceito de *embedded ioctls*, no qual em apenas uma `ioctl` (genérica e implementada na camada VFS) é passado um argumento, o qual contém um conjunto de uma ou mais `ioctl` criadas a partir de um *bitwise* OR entre ioctls GoboHide.

Além da adição/remoção/consulta de *inode numbers*, é ainda fornecido ao usuário uma *ioctl* para traduzir os *inode numbers* armazenados na estrutura GoboHide para o *pathname* completo dos diretórios e/ou *links* simbólicos mantidos nesta estrutura.

4. Administração da Árvore *legacy* em espaço de usuário

Como a implementação GoboHide no Kernel Linux cria um *framework* para gerência da árvore clássica do GNU/Linux, foi necessário desenvolver uma aplicação em espaço de usuário para utilizar as operações disponíveis pelo GoboHide-Kernel.

A ferramenta fornece ao administrador do sistema uma interface simples e transparente para ocultar diretórios e/ou *links* simbólicos da árvore *legacy* do GNU/Linux. Ela possui uma CLI (*Command Line Interface*) com as seguintes opções:

```
$ gobohide --help

gobohide: Hide/Unhide a directory

-h, --hide     Hide the directory
-u, --unhide   Unhide the directory
-l, --list     List the hidden directories
    --version  Show the program version
```

```
--help      Show this message
```

A ferramenta basicamente identifica (utilizando a chamada de sistema `stat`) o *inode number* do diretório e/ou link simbólico que o administrador do sistema deseja ocultar e cria a devida GoboHide Embedded ioctl, utilizando a ioctl genérica FIGOBOLINUX (que é a ioctl genérica das estruturas GoboHide) para comunicação com o GoboHide-Kernel.

Tendo acrescentado, por exemplo, o diretório */etc* à lista de diretórios ocultos pelo GoboHide, o administrador do sistema **não** precisa removê-lo (utilizando a opção '-u') da lista de ocultos caso ele seja removido do sistema, pois o GoboHide-kernel faz a detecção automática pela modificação das chamadas de sistemas `unlink` e `rmdir`, e transparentemente na próxima consulta às estatísticas do GoboHide o administrador irá notar que a lista atual possui apenas entradas consistentes (não desperdiçando memória de kernel).

As "estatísticas", que informam ao administrador do sistema quais diretórios estão sendo escondidos pelo GoboHide, são alocadas em kernel, porém em contexto de usuário, ou seja: a memória utilizada para alocar as estruturas necessárias para as estatísticas é obtida utilizando memória virtual do usuário, não fazendo uso da memória de kernel para isso.

5. Conclusões e Trabalhos Futuros

Através da utilização do GoboHide é possível manter compatibilidade de uma maneira simples e consistente entre uma nova árvore de diretórios e a árvore clássica do sistema operacional UNIX, sem que a árvore de diretórios clássica fique visível para o usuário, como é visto na distribuição GoboLinux[Muhammad and Detsch, 2002].

Como a implementação do GoboHide está em kernel, porém com uma interface administrativa em espaço de usuário, existe uma transparência para qualquer processo da árvore clássica, porém programas antigos com *paths hardcoded* baseados na árvore clássica funcionam corretamente, devido à não modificação nas chamadas de sistema `open` e `close`.

O *design* do GoboHide em kernel foi feito de forma a possibilitar, através do uso de *embedded ioctls*, a adição de novas operações ao framework de operações suportadas pela ferramenta administrativa em espaço de usuário, permitindo que outras distribuições baseadas no kernel Linux possam usufruir dessas operações.

Nota-se ainda que como o GoboHide precisa acessar as estruturas dependentes de sistema de arquivos para interceptar o *filldir* na chamada de sistema *readdir*, nem todos os sistemas de arquivos suportados pelo kernel Linux são suportados pelo GoboHide. Uma possível solução para esse problema é a utilização de *Stackable File Systems*[Heidemann, 1995], no qual é acrescentando mais um nível de indireção na camada VFS, no qual seria possível criar um sistema de arquivos "virtual" chamado GoboHide que interceptaria a chamada de sistema `readdir` e passaria para o sistema de arquivos específico apenas os inode numbers que não devem ser ocultados.

Referências

Apple Computer Inc. (2001). *Inside Mac OS X - System Overview*. ISBN: 1400524806.

Heidemann, J. S. (1995). Stackable Design of File Systems. Technical report, University of California, Los Angeles. Ph.D. dissertation.

Muhammad, H. and Detsch, A. (2002). Uma nova proposta para a árvore de diretórios UNIX. Proceedings of the III WSL - Workshop em Software Livre.

Autenticação de Senhas Faladas Dependente de Usuário

Lucas Correia Villa Real*
Farlei José Heinen
Luiz Paulo Luna de Oliveira

[1]Programa de Pós-Graduação em Computação Aplicada – PIPCA
Centro de Ciências Tecnológicas e Exatas
Universidade do Vale do Rio dos Sinos
São Leopoldo – RS – Brasil

{lucasvr, farlei, luna}@exatas.unisinos.br

Abstract. *This paper presents* VoiceAuth[1]*, a system developed to allow voice authentication for GNU/Linux. By using it, the user is recognized based on the similarity between a previously spoken reference (and stored) password, and a new speech of the same password. The comparison is done using correlation analysis between the coefficients taken from the Windowed Fast Fourier Transform of both passwords (reference and the new speech).*

Resumo. *Este artigo apresenta* VoiceAuth[1]*, um sistema para permitir autenticação de usuários a partir de senhas faladas no GNU/Linux. No seu uso, o usuário é reconhecido pela comparação entre uma senha falada de referência previamente adotada (e armazenada), e uma nova execução (pronúncia) da mesma senha. A comparação se baseia na correlação entre os coeficientes obtidos através da Transformada Rápida de Fourier janelada de ambas as senhas (de referência e executada).*

1. Introdução

O uso de métodos biométricos para autenticação permite que o usuário tenha uma maneira mais natural de garantir a sua identidade. Alguns destes métodos incluem reconhecimento através da íris, de impressões digitais, reconhecimento da face e reconhecimento da voz [Matyas and Riha, 2000]. Um sistema deste tipo é desejável nos dias atuais, onde sistemas de autenticação baseados em biometria são utilizados para aumentar o acesso seguro a dispositivos, além de permitir a pessoas deficientes uma interface efetiva para realizar acessos a suas contas de usuário ou a outros recursos com uso de senhas. É ainda importante destacar que, pela disponibilidade do GNU/Linux, o software proposto pode ser utilizado em um largo conjunto de sistemas embarcados, além das estações de trabalhos e computadores pessoais.

A escolha de um método baseado em reconhecimento de voz tem, como motivação, a disponibilidade do hardware necessário em uma grande parte dos computadores atuais (microfones, placas de som, etc.). Isto permite que o sistema seja utilizado de forma mais ampla do que caso fosse necessário o uso de hardware caro e restrito. Neste trabalho, propomos um método de reconhecimento de usuário por senha falada dependente de locutor, com o uso de software livre e bibliotecas de grande generalidade.

*ITI-CNPq

[1]http://cscience.org/ lucasvr/projects/voiceauth.php

Uma senha falada pode se constituir de uma palavra ou frase que, em última análise, é uma seqüência temporal de sons específicos. Por isto, é desejável que um método de reconhecimento de senhas faladas seja sensível tanto às mudanças temporais dos sons quanto ao seu conteúdo de freqüências. Por outro lado, é desejável também que ele seja sensível ao locutor, o que se traduz novamente em uma sensibilidade do método aos conteúdos de freqüências. Motivados por isso, propomos um método que se baseia na comparação por correlação dos coeficientes da Transformada Rápida de Fourier (FFT) janelada [Oppenheim and Schafer, 1980], o que dá a este método a característica de ser sensível ao vocábulo propriamente dito (sílabas e suas durações) bem como ao locutor, com o conteúdo de freqüências características de sua voz.

2. Método Proposto

VoiceAuth foi desenvolvido como um módulo para a biblioteca Linux-PAM (*Pluggable Authentication Modules*), estendendo as funcionalidades desta. O Linux-PAM é uma implementação de software livre baseada na proposta original do PAM da SUN Microsystems [Samar and Schemers, 1995]. O Linux-PAM é uma biblioteca que fornece um mecanismo flexível para a autenticação de usuários. Utilizando PAM, não há a necessidade de modificar o código fonte de aplicações para que estas façam o uso deste método biométrico (desde que elas já tenham suporte às facilidades oferecidas pelo PAM). Assim, tem-se um método de autenticação que permite a uma grande gama de aplicativos aproveitar-se deste recurso.

No seu uso, o usuário pronuncia pelo microfone uma senha falada de referência, que então é pré-processada pela Transformada de Fourier Janelada. São então gravados os coeficientes de Fourier de cada uma das J janelas de Fourier utilizadas. Para autenticar seu login, o usuário executa novamente a senha de referência, falando-a pelo microfone. Da execução da senha, que será então pré-processada da mesma maneira que a senha de referência, extraem-se os coeficientes de Fourier de cada janela, os quais serão comparados por correlação com aqueles da senha de referência, armazenados anteriormente. Abaixo, são dados mais detalhes do uso do método proposto.

Registro da senha de referência:

- **Calibragem de Silêncio:** usada para determinar o nível de ruído do ambiente. O conjunto de dados que será processado pelo software inicia após a primeira ocorrência de uma amplitude superior ao nível de silêncio determinado pelo sistema, e termina quando este é reestabelecido. Tal calibragem atualmente dura 3 segundos.
- **Registro da Senha:** o usuário expressa sua senha através da voz quando solicitado pelo software. Fazendo uso da amplitude definida na etapa de calibragem de silêncio, apenas o sinal relevante é processado pelo software.
- **Armazenamento Físico:** o nome do usuário é adicionado a uma entrada em um arquivo, onde têm-se controle de quais usuários estão fazendo uso deste sistema de autenticação biométrico. Um arquivo contendo as partes *reais* dos coeficientes de Fourier da senha nas J janelas utilizadas é armazenado para futuras comparações com as tentativas de autenticação no sistema.

Entrada no sistema (login):

- **Calibragem de Silêncio:** bem como no registro da senha de referência, este passo é necessário para determinar o início e o término da fala do usuário.
- **Tentativa de Autenticação:** o usuário expressa através da voz sua senha previamente registrada.
- **Normalização do Sinal:** como o sistema precisa comparar a senha falada com a senha armazenada, os sinais precisam estar representados na mesma escala. Este passo realiza

a normalização do sinal em função do tempo (comprimento dos sinais) e da amplitude (intensidade do sinal). Caso o comprimento dos sinais tenha uma diferença significativa (onde esta diferença pode ser especificada pelo administrador), o processamento é cancelado neste ponto, e a autenticação não é bem sucedida pelo usuário.

- **FFT Janelada:** a senha falada é dividida em J janelas. O final do *buffer* referente às janelas é preenchido com *zeros*, caso o sinal não ocupe-o completamente. Uma rotina de FFT é aplicada nestas J janelas, modificando a representação do sinal: ele passa da representação em função do tempo para uma representação em função das freqüências, permitindo obter dados relativos à identidade do locutor.
- **Correlação:** após processadas pela FFT, as freqüências são analisadas em relação à similaridade das partes *reais* de seus coeficientes. Para isto, uma função de correlação é aplicada na saída gerada pela FFT para cada uma das J *janelas*. Existe um limite X definido pelo módulo em um arquivo de configuração: caso 2 janelas subseqüentes tenham uma correlação inferior a X, as senhas são definidas como diferentes, e o reconhecimento é interrompido. Caso não hajam janelas subseqüentes com valores com valores inferiores a este limite X estabelecido, a média das somas obtidas no processo de correlação para as J janelas é comparada com um outro limite Y. Caso esta média seja maior que o valor especificado por Y, as senhas são ditas equivalentes, e o usuário é autenticado.

3. Interfaces

Para o uso deste sistema, alguns programas foram desenvolvidos para auxiliar a administração do sistema. O módulo contendo as rotinas de autenticação do PAM é apresentado na biblioteca compartilhada `voiceauth.so`. Para validar este módulo, um programa denominado `voicepasswd` foi criado, de forma a manter um registro de usuários que têm permissão para o uso deste serviço. Para todo usuário adicionado a este arquivo, um arquivo extra, salvo sob o mesmo nome do *username*, é gerado contendo a senha deste usuário. Um programa denominado `voicelogin` foi ainda criado para interagir com o módulo, permitindo que o login padrão do sistema seja realizado através da voz.

Existem várias configurações do sistema que podem ser ajustadas para modificar dispositivos e características na função de correlação. Os parâmetros aceitos pelo módulo são:

- `va_dir:` define o diretório a ser utilizado como repositório para os arquivos de senha;
- `audiodev:` define o dispositivo de áudio a ser utilizado, ao invés do padrão `/dev/dsp`;
- `mixerdev:` define o dispositivo de mixer a ser utilizado, ao invés do padrão `/dev/mixer`;
- `mic_level`: ajusta o volume do microfone a um nível específico;
- `igain_level:` ajusta o ganho no sinal de entrada a um nível específico;
- `silence_level:` ajusta o nível de silêncio. Este parâmetro pode ser utilizado em um ambiente onde o nível de silêncio é conhecido. Este valor pode ainda ser redefinido pelo aplicativo, no instante da calibragem do nível de silêncio do ambiente;
- `silence_threshold:` define a amplitude na qual o sistema irá passar a tratar o sinal, definindo o início e término da fala do locutor;
- `corr_threshold:` determina o grau de semelhança necessário entre os sinais para defini-los como “iguais” ou não.
- `passwd_maxlen_diff:` determina a diferença máxima permitida no tamanho entre a senha armazenada e a falada. Caso uma senha tenha um comprimento fora dos limites estabelecidos, ela já é dada como incorreta, poupando o processamento da FFT.

VoiceAuth utiliza-se das bibliotecas OSS e ALSA para interagir com o dispositivo de som. A OSS (*Open Sound System*) é a interface original para a programação de áudio no GNU/Linux,

mantido pela 4Front Technologies. A OSS suporta a maioria das placas de som atuais, no entanto alguns dos drivers mais recentes necessitam de uma licença comercial da OSS. O ALSA (*Advanced Linux Sound Architecture*) é um projeto comunitário que busca superar o OSS em todas as áreas, fornecendo um conjunto completo de drivers de placas de som, livres de licenças comerciais.

A freqüência de amostragem utilizada no VoiceAuth é de 44100 Hz, enquanto a quantização (8, 16, 24-bit) é determinada pelo módulo `voiceauth.so`, permitindo que a melhor qualidade de áudio seja empregada [Pholmann, 1995].

4. Resultados e Conclusão

O recurso apresentado neste artigo consiste em uma extensão de uma ferramenta disponibilizada segundo a política de Software Livre. O sistema propõe ser uma alternativa válida como método de autenticação para sistemas GNU/Linux, tendo como vantagem a implantação de um sistema de autenticação biométrico de baixo custo. Os resultados preliminares obtidos com os testes demonstraram que o sistema foi capaz de identificar usuários através da senha falada, com uma taxa de acerto não inferior a 80%, apesar de um pequeno tempo de treinamento ser necessário para que o usuário consiga habituar-se a expressar sua senha através da voz. Nos testes preliminares não foram utilizados equipamentos profissionais, como hardware dotado de filtros ou de conversores de alta fidelidade.

Ainda existe a possibilidade de adaptar um método de criptografia às senhas armazenadas no sistema. Entretanto, um certo grau de segurança já é garantido pelo armazenamento da senha a partir da parte *real* dos coeficientes de Fourier. Isto permite que sejam armazenados apenas os dados relevantes à função de correlação, evitando que um possível invasor no sistema tenha acesso ao sinal original. Na implementação atual, existe ainda uma precaução relacionada a evitar que a memória utilizada pela leitura da senha seja *swapped-out* para o disco rígido, através da família de funções *mlock()*. Além disso, as permissões utilizadas pelos arquivos de senhas individuais são feitas de forma a garantir que estes sejam lidos apenas pelos seus proprietários.

A idéia apresentada admite muitas possibilidades de aperfeiçoamento. Por exemplo, pode-se tentar separar os conteúdos de freqüência do locutor (*pitch*) e da senha propriamente dita. Feito isto, a validação de senhas poderia se dividir em reconhecimento de locutor e da senha propriamente dita. Isto poderia agilizar o processo e até mesmo torná-lo mais preciso. Outra possível melhoria seria a implementação do método utilizando técnicas de Wavelets para o processamento de sinais. Tais extensões estão sendo consideradas como temas de trabalhos futuros.

Referências

Matyas, V. and Riha, Z. (2000). Biometric Authentication — Security And Usability.

Oppenheim, A. V. and Schafer, R. W. (1980). *Discrete - Time Signal Processing*. Prentice-Hall International, Inc.

Pholmann, K. (1995). *Principles of Digital Audio*. McGraw-Hill, Inc., 3 edition.

Samar, V. and Schemers, R. (1995). Request for comments: Unified Login with Pluggable Authentications Modules (PAM). Technical Report 86.0, SunSoft.

DiretoJ2EE: uma Modelagem J2EE para o Direto

Karina K. Silveira[1] **, Sérgio L. B. Vidal**[1] **, Flávio R. Maciel**[1] **, Rômulo B. Rosinha**[1] **, Cláudio F. R. Geyer**[1]

[1] Instituto de Informática – Universidade Federal do Rio Grande do Sul
Av. Bento Gonçalves, 9500 - Campus do Vale - Bloco IV
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

{kohl,vidal,frm,romulo,geyer}@inf.ufrgs.br

Abstract. *This article describes the studies about a J2EE implementation for Direto, the free, open source enterprise mail, scheduling and directory software developed by PROCERGS. The goal of these studies is to achieve better modular, reusable and scalable software.*

Resumo. *Este artigo descreve os estudos de implementação da arquitetura do Direto, software livre de correio, agenda e catálogo corporativos desenvolvido pela PROCERGS, para a especificação Java 2, Enterprise Edition (J2EE). Estes estudos foram desenvolvidos visando obter melhor modularidade, reusabilidade e escalabilidade ao software em questão.*

1. O Direto

O Direto [DiretoGNU, 2003] é um *software* de agenda, catálogo e correio corporativo desenvolvido pela PROCERGS, Companhia de Processamento de Dados do Estado do Rio Grande do Sul, com o objetivo de atender os diversos órgãos do estado. A PROCERGS optou pelo desenvolvimento interno de uma ferramenta de *groupware*, pois dessa forma seria permitida a evolução gradual e atendimento das necessidades especificas dos usuários. Além disso, o desenvolvimento de uma solução própria é de grande atrativo em relação a custos e flexibilidade da solução. Esse baixo custo justifica-se pelo Direto possuir licença de *software* livre e basear-se totalmente em tecnologias baseadas na filosofia de *software* livre, tais como Java, Apache, Tomcat, OpenLDAP e PostgreSQL.

Porém, como constatado em estudos já realizados [Geyer et al., 2001], o sistema peca na questão da modularidade, por não usar da melhor forma o paradigma orientado a objeto do qual tem poder com o uso da linguagem Java. Os componentes de *software* do Direto possuem um alto grau de acoplamento entre si, o que torna a sua manutenção difícil e o prejudica em relação à escalabilidade [Maciel, 2002].

A partir da proposta de portar o Direto para o ambiente J2EE [Sun Microsystems, 2003a], alguns trabalhos foram realizados, como a modelagem e prototipação da Agenda [Rosinha, 2002] [Silveira, 2003], Catálogo [Maciel, 2002] e Correio [Vidal, 2003]. Este trabalho apresenta alguns detalhes sobre as tecnologias utilizadas e modelagens realizadas.

2. As Tecnologias

A plataforma J2EE é um conjunto de tecnologias de suporte ao desenvolvimento de aplicações corporativas distribuídas. Ela implementa serviços de baixo nível que tendem a facilitar a reutilização

de componentes em diferentes aplicações e que permitem ao desenvolvedor concentrar-se principalmente na implementação da lógica de negócio. Tais características da arquitetura J2EE vão ao encontro das necessidades do Direto, e, por isso, foram propostas soluções aos problemas do Direto, baseadas em J2EE, para os módulos de Agenda, Catálogo e Correio.

O JavaMail [Sun Microsystems, 2003b] é o *framework* de correio eletrônico para aplicações Java. Essa API foi projetada visando disponibilizar funcionalidades de correio eletrônico de forma independente de plataformas e protocolos através de uma interface simples e leve, e ao mesmo tempo suportar a criação de interfaces do usuário sofisticadas. Essa API foi utilizada para a proposta de um novo módulo de Correio.

A tecnologia JDO [Sun Microsystems, 2003c], *Java Data Objects*, é uma definição baseada em interface para a persistência de objetos. Essa definição descreve o armazenamento, consulta e recuperação de objetos de bases de dados. Utilizou-se essa tecnologia em conjunto com os Enterprise Java Beans da plataforma J2EE, propondo uma nova forma de persistir os eventos da Agenda de um usuário.

O JBoss [JBoss Group LLC, 2003] é um servidor de aplicações J2EE de código livre, totalmente escrito em Java, porém não certificado oficialmente pela Sun. Em 2002 chegou a 200 mil *downloads* mensais, se tornando o servidor J2EE com maior número de *downloads*, e foi eleito como o melhor servidor de aplicação pela revista online JavaWorld, concorrendo com o BEA, IBM, Sun e outros. O fato do JBoss ser distribuído sob a licença LGPL permite a sua utilização de forma comercial. Por exemplo, é possível distribuir uma aplicação baseada em J2EE em um CD junto com o JBoss, utilizá-lo em conjunto com outras aplicações comerciais, ou até mesmo alterar seu código fonte gerando um novo produto. Além disso, o uso de software gratuito reduz o custo total de propriedade [Gomes, 2002].

3. O Direto e J2EE

Foram propostas quatro modelagens baseadas nas tecnologias apresentadas, para o software Direto. Duas delas são para o módulo Agenda, a primeira baseada em *entity bean*s para persistência do dados [Rosinha, 2002] e a segunda baseada em JDO para persistência [Silveira, 2003]. Também foi proposto um modelo para o módulo de Catálogo, utilizando o protocolo LDAP e também um modelo para o módulo de Correio, utilizando-se *session beans* e a API Java Mail.

Os padrões de projeto de *session beans* de Fachada e de Objetos de Transferência foram amplamente utilizados nos trabalhos e merecem atenção especial.

3.1. Beans de Fachada

Os *session beans* geralmente tem a função de apresentar uma interface unificada a clientes fora do contêiner EJB, para a manipulação de um conjunto de *entity beans*. Para este tipo de função é dado o nome de fachada, por omitir de seu cliente o uso de *entity beans* internos [Maciel, 2002]. O bean de fachada permite o acesso aos serviços oferecidos pelo contêiner EJB. Além disso, como somente ele pode ser acessado remotamente pela camada Web, ele se torna o único ponto de entrada para o modelo da aplicação. Isso resulta em baixo acoplamento entre a interface e o modelo da aplicação [Vidal, 2003].

3.2. Objetos de Transferência

Para a execução dos casos de uso dos modelos que serão apresentados, é necessário que o *session bean* de fachada transmita e/ou receba dados à camada Web de maneira eficiente, usando o menor número de chamadas remotas. Ao mesmo tempo, o *session bean* não deve expor os objetos do

domínio para a camada Web, pois isso prejudicaria a independência entre essas camadas que o *session bean* de fachada garantiria. Para obter uma comunicação dos dados entre as camadas Web e EJB usando uma única chamada remota sem estabelecer dependências entre os objetos da interface e do domínio, os dados são encapsulados em objetos de transferência (*transfer objects*), classes Java que encapsulam os dados a serem transmitidos e são serializáveis, isto é podem ser transformadas em fluxos de dados para comunicação remota [Vidal, 2003].

4. As modelagens Propostas

A seguir são apresentadas as modelagens propostas aos módulos de Agenda, Catálogo e Correio, utilizando-se J2EE em conjunto com outras tecnologias baseadas em software livre.

4.1. O Módulo Agenda

Dois trabalhos foram realizados no intuito de propor modelos J2EE para o módulo de agenda do Direto. O primeiro [Maciel, 2002], baseou-se em persistência com *entity beans* para os eventos de agenda. O segundo [Silveira, 2003], baseou-se em um novo padrão proposto pela SUN para persistência de objetos, o JDO.

Em ambas as modelagens utilizou-se *stateless session beans*, como beans de fachada, pois a análise dos métodos da agenda revelou que a natureza dos métodos expostos é transacional, ou seja, não existe real necessidade de se manter o estado de um *session bean* e mantê-lo alocado exclusivamente para cada usuário acessando o sistema para manter a sua sessão.

Os dois trabalhos realizados para a agenda diferenciam-se em sua camada de persistência. O primeiro trabalho [Rosinha, 2002] utiliza um *entity bean* que representa a entidade evento do domínio do problema. Ele usa persistência *container-managed* (gerenciada pelo contêiner EJB).

Para a modelagem usando-se JDO para persistência dos dados da Agenda [Silveira, 2003], além dos EJBs é necessária a criação de uma classe que será a classe a ser persistida. Essa classe não difere de uma classe Java comum e pelos conceitos JDO é chamada de *PersistenceCapable*, ou seja, uma classe que pode ser persistida. Nesse caso, a utilização de um *entity bean* seria redundante, pois *entity beans* e JDO são utilizados para persistência.

4.2. O Módulo Catálogo

Para a modelagem do catálogo [Maciel, 2002], identificou-se que as entidades que modelam os dados que serão persistidos são grupo e contato e foram modeladas como *entity beans*. A entidade catálogo provê o meio pelo qual os dados de grupo e contato são manipulados, portanto, foi modelada como um *session bean* de fachada.

Para o caso do catálogo, utilizou-se um objeto de transferência para grupo e outro para contato. Quando for requisitado ao bean de fachada os contatos do catálogo pessoal, esse obtém as referências a todos os contatos, encapsula os dados de todos estes objetos em objetos de transferência e os envia para o cliente.

4.3. O Módulo Correio

Embora *entity beans* sejam o componente padrão da especificação J2EE para a modelagem dos objetos do domínio, a modelagem proposta para o módulo de correio não implementa suas classes dessa forma. Esta abordagem, que difere daquelas empregadas nos módulos da Agenda [Rosinha, 2002] [Silveira, 2003] e do Catálogo [Maciel, 2002], deve-se a dois fatores. Primeiro, a persistência gerenciada pelo contêiner que o J2EE oferece é aplicável apenas a banco de dados relacionais; uma modelagem de mensagens e pastas de correio como *entity beans* necessariamente usaria persistência gerenciada pelo bean, eliminando uma das principais vantagens dos

entity beans, que é a persistência transparente gerenciada pelo contêiner. Assim sendo, as classes Mensagens e Pasta não utilizam os recursos dos *entity beans*, podendo ser implementadas como *session beans* ou classes Java simples [Vidal, 2003].

Por outro lado, sendo os objetos de domínio implementados como classes Java simples, eles não tem acesso aos serviços oferecidos pelo contêiner EJB, como controle de transações e de acesso. Para se aproveitar dessas características inerentes aos EJBs, foi necessária a implementação de um *session bean* de fachada para a camada de domínio.

5. Conclusões

A utilização de tecnologias baseadas em software livre é uma ótima escolha para o desenvolvimento de sistemas, pois oferece baixo custo e uma grande liberdade de customização de sistemas de uma empresa. Além disso, apresentou-se algumas soluções baseadas na plataforma J2EE da Sun, para o *software* Direto da PROCERGS, valendo-se da arquitetura em camadas e do fato da tecnologia Java ser livre e poder ser utilizada com tantas outras tecnologias de mesmo filosofia. A arquitetura de camadas imposta pelo J2EE permite a remodelagem do sistema utilizando baixo acoplamento e maior modularidade entre os módulos do Direto.

Referências

DiretoGNU (2003). DiretoGNU - Comunicação Objetiva. Disponível em http://www.direto.org.br.

Geyer, C. F. R. et al. (2001). Projeto Sistemas Avançados para Comunicação Eletrônica - Software Aberto de Correio, Agenda e Catálogo. In *Anais do 2o Workshop sobre Software Livre*, pages 25–28.

Gomes, H. (2002). JBoss 3. *Revista Java Magazine*. Edição 2, Ano 1. Dezembro 2002.

JBoss Group LLC (2003). JBoss. Disponível em: http://www.jboss.org. Acesso em: 06 março 2003.

Maciel, F. R. (2002). Modelagem do Catálogo e Autenticação do Direto utilizando J2EE e JAAS. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Rosinha, R. B. (2002). DiretoJ2EE: Modelagem e Prototipação do Direto para a Plataforma J2EE. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Silveira, K. K. (2003). DiretoJ2EE: Uso de JDO para Persistência no Módulo Agenda. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Sun Microsystems (2003a). J2EE. Disponível em http://java.sun.com/j2ee/index.html. Acesso em: 06 março 2003.

Sun Microsystems (2003b). JavaMail(TM) API v 1.2. Disponível em http://java.sun.com/products/javamail/index.html. Acesso em: 06 março 2003.

Sun Microsystems (2003c). JDO v 1.2. Disponível em http://java.sun.com/products/jdo/index.html. Acesso em: 06 março 2003.

Vidal, S. L. B. (2003). Modelagem J2EE do Módulo Correio do Direto. Projeto de diplomação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Módulo Agenda do DiretoGNU: Reestruturação das Classes e Adaptação das Interfaces

Alex Francisco de Oliveira[1], Débora Nice Ferrari[1,2]
Patrícia Kayser Vargas Mangan[1,3], Cláudio Fernando Resin Geyer[2]

[1]Centro Universitário La Salle (UNILASALLE)
Av. Victor Barreto, 2.288 – 92.010-000 – Canoas, RS, Brasil

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre, RS, Brasil

[3]COPPE/Sistemas – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 68.511 - 21941-972 – Rio de Janeiro, RJ, Brasil

alexfgo@bol.com.br, {nice, kayser}@lasalle.tche.br,
kayser@cos.ufrj.br, {nice,geyer}@inf.ufrgs.br

Abstract. *The DiretoGNU is a free software of agenda, mail and catalog. This work presents the adaptation and restructuring of DiretoGNU's Agenda Module. We present all modified classes and methods and its relationship, and a brief description of the functionalities obtained in this modification process. We also show the applied design patterns and the new interfaces defined. Besides, a new module of free task list was proposed and implemented adding new functionalities to DiretoGNU's Agenda Module.*

Resumo. *O DiretoGNU é um software livre de agenda, correio e catálogo. Este trabalho apresenta a adaptação e reestruturação do Módulo Agenda do DiretoGNU. Destaca-se a relação de todas classes e métodos modificados, além de uma breve descrição das funcionalidades obtidas. São expostos também os padrões de projeto utilizados e a forma como as novas interfaces foram definidas. Além disto, um módulo de lista livre de tarefas foi proposto e implementado, de forma a adicionar facilidades a Agenda do DiretoGNU.*

1. Introdução

Sistemas computacionais costumam possuir complexidade proporcional ao conjunto de funcionalidades disponibilizadas. Na prática é usual a construção de um sistema simples no qual novas funcionalidades são adicionadas aumentando a sua complexidade. Para permitir que esta complexidade seja gerenciável, a adoção de boas práticas de engenharia de software permite facilitar a detecção de erros e a inclusão de funcionalidades adicionais. O ideal seria pensar desde a concepção inicial (fase de projeto) nestas questões.

Neste sentido, o aperfeiçoamento do Módulo Agenda do DiretoGNU1, que é um software relativamente complexo, serve como uma forma de ilustrar a busca da melhor forma de aliar a gerência de crescimento de código e aproveitamento de código legado baseando-se em duas práticas: (1) Redefinição de interfaces: criar um padrão para acesso às funcionalidades do módulo, aproveitando o código existente; (2) Reestruturação das classes: facilitar o

[1]http://www.direto.org.br

entendimento da atividade alvo do aplicativo e permitir que manutenções e expansões possam ser feitas de forma mais rápida e menos complicada, diminuindo o grau de conhecimento prévio do sistema para realização da tarefa. Estas premissas foram levadas em consideração no aperfeiçoamento do módulo de Agenda do DiretoGNU que este trabalho apresenta. Além disso, foi desenvolvido um novo conjunto de funcionalidades para Gerenciamento de Tarefas Livres, isto é, tarefas que possuem um prazo indeterminado ou mais flexível para a sua conclusão, adicionando mais facilidades de gerência do tempo ao sistema.

Deste modo, o restante do texto está dividido da seguinte forma: a seção 2 apresenta o processo de reestruturação do módulo de agenda e a criação da lista de tarefas livres. Na seção 3 são apresentas as considerações finais.

2. Reestruturação das Classes e Adaptação das Interfaces

Nos tópicos que seguem é apresentada à metodologia adotada para realização do projeto e sua utilização no trabalho desenvolvido.

2.1. Metodologia adotada

A escolha da metodologia levou em consideração aspectos relacionados à necessidade de aproveitamento do código legado, tempo e ambiente disponível. Para conciliar estes três aspectos a opção natural foi à adoção de Padrões de Projeto (*Design Patterns*) [1] apropriados. O principal padrão escolhido foi o Adaptador (*Adapter*). Com isso, o modelo de classes foi reestruturado e as interfaces adaptadas de forma a acessar o código legado. Uma visão geral da adaptação de interfaces e reestruturação de classes pode ser vista na figura 1 abaixo:

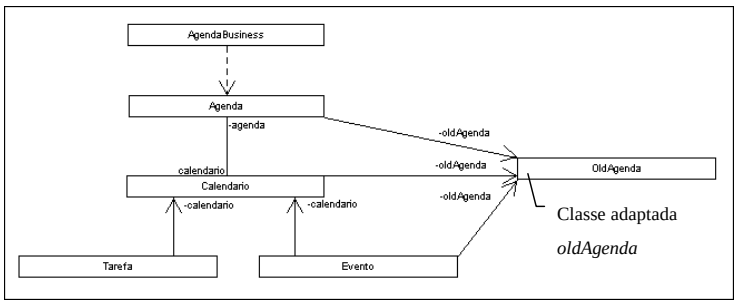


Figura 1. Diagrama de Adaptação e Reestruturação do Módulo de Agenda

Neste diagrama aparecem as novas classes geradas a partir da reestruturação do modelo, isto é, classes *Agenda*, *Calendario*, *Evento* e *Tarefa*. Também aparece a antiga classe contendo o código legado (*oldAgenda*) na qual os métodos originais continuam sendo executados. Este é um exemplo típico de **adaptação por instância**, na qual cada uma das classes possui uma ligação com a classe adaptada (*oldAgenda*) para aproveitamento dos métodos legados [1]. A adaptação poderia ser feita por herança especializando os métodos antigos e alterando suas interfaces. Entretanto, a opção por dividi-la facilitou fundamentalmente o entendimento e o isolamento do código, o que é de extrema importância para manutenções futuras. Além disso, ocorreu também a criação de uma nova classe para gerenciamento de tarefas, visando complementar o sistema com funções adicionais para organização do tempo conforme será discutido na seção 3.

2.2. Alterações de código e criação da lista livre de Tarefas

As modificações realizadas no código procuraram manter ao máximo o código original do projeto, não exigindo modificações em classes que não fizessem parte do módulo Agenda. A exceção da classe *AgendaBusiness*, interface responsável pela invocação dos métodos da *Agenda*, nenhuma outra classe original teve seu código modificado. A classe *AgendaBusiness*, por fazer referência direta ao código da *Agenda*[2] teve seus métodos modificados para realizar a passagem correta dos parâmetros para as novas classes criadas a partir de *oldAgenda*[3].

A divisão das funcionalidades da classe *oldAgenda* em novas classes menores e mais especializadas constitui a maior alteração do código. Basicamente, foram reunidas em cada uma das novas classes métodos e atributos relacionados mais diretamente a cada uma destas partes. Um exemplo pode ser dado por um *Evento*: agora ele possui métodos para *salvar, excluir, marcar, remarcar* e *delegar*, que são chamados a partir de uma classe *Calendario*, responsável por gerenciar todos os eventos que pertencem à agenda de um determinado usuário. Assim, cria-se um nível de isolamento permitindo um melhor tratamento de erros e detecção de problemas. Anteriormente, todos os métodos pertenciam à mesma classe (*oldAgenda*), juntamente com todos os demais métodos pertinentes ao gerenciamento das funcionalidades do módulo. Outra grande vantagem deste modelo de adaptação é permitir que, ao longo do tempo, métodos legados possam ser migrados para dentro das novas classes sem interferir no funcionamento do módulo, que permanece enquanto for necessário, utilizando o código legado. Esta funcionalidade é obtida em função da padronização das interfaces. As principais modificações realizadas nas classes são especificadas a seguir. Inclui-se também a classe *Tarefa* especialmente criada para contemplar funcionalidades inexistentes quanto a lista livre de tarefas.

Classe AgendaBusiness

Esta classe funciona na prática como uma interface para a principal classe do módulo: *Agenda*. As principais mudanças ocorridas foram causadas pela necessidade de adaptação das assinaturas dos métodos das novas classes em relação ao código legado. Ao mesmo tempo, foi necessário manter também a compatibilidade com as chamadas realizadas pela interface gráfica, evitando com isso a modificação do código nas interfaces existentes [4]. Todos os métodos da classe tiveram suas assinaturas modificadas, à exceção do método público *indice*, que precisou ser mantido por motivos de compatibilidade. Este é o principal método da classe, responsável por redirecionar as chamadas e parâmetros enviados para os métodos específicos da instância do objeto *agenda*. A assinatura original foi mantida para que a compatibilidade com as interfaces existentes não fosse prejudicada, mas suas chamadas internas foram completamente modificadas pela alteração dos nomes dos métodos privados, de forma a dar mais clareza e precisão quando a sua própria utilidade e, além disso, para criação de um padrão de chamadas para o novo conjunto de classes. Por sua vez, estes métodos privados também sofreram modificações internas visando adaptar os parâmetros recebidos as assinaturas dos métodos criados nas novas classes (*Agenda*, *Calendario*, *Evento* e *Tarefa*).

Classe OldAgenda (antiga classe Agenda)

Nesta classe está concentrada a maior parte do código utilizado no módulo Agenda. Por ser consideravelmente grande, tornando-a complexa e difícil de manter, foi mantida na sua forma original, tendo suas chamadas mapeadas em três classes criadas e organizadas de forma a aproveitar o código legado e tornar sua compreensão mais fácil. As classes são as seguintes: (A) **Classe Agenda** - resultado da otimização da classe *OldAgenda*. Reúne basicamente todos os

[2] Nova classe *Agenda* criada a partir da divisão da antiga classe que também chamava-se Agenda.

[3] Antiga classe Agenda que para fins de adaptação foi renomeada para *oldAgenda*.

métodos estáticos (métodos de classe) existentes na antiga classe, renomeados para criação de um padrão de assinaturas. Não possui nenhum código novo, excetuando as chamadas para criação de objetos internos como *calendario*. Cada objeto agenda possui como objeto interno um outro da classe *Calendario.* Esse possui ainda alguns métodos intermediários que simplesmente repassam chamadas para este objeto interno. Estes métodos foram criados prevendo possíveis expansões do sistema ou migração do código legado; (B) **Classe Calendario -** classe instanciada sempre a partir da classe *Agenda*, possuindo métodos relacionados com a manutenção do conjunto de eventos do usuário. Realiza as chamadas diretamente ao código legado existente no objeto interno o*ldAgenda*. Este objeto é atribuído ao calendário no momento de sua criação por sua classe mãe: *Agenda*. Não possui métodos estáticos e utiliza a conexão estabelecida pela classe *Agenda* com o banco de dados; (C) **Classe Evento -** reúne as chamadas ao código legado relacionadas diretamente com a criação, exclusão, edição, aceitação, rejeição, importação e exportação de eventos ou compromissos, como eram chamados na sua forma antiga.

Classe Tarefa

Uma nova classe denominada *Tarefa* foi especialmente criada para contemplar funcionalidades inexistentes, isto é, criar e gerenciar a lista livre de tarefas. Assim, a classe *Tarefas* foi criada utilizando as práticas da orientação a objetos e totalmente integrada as demais classes existentes evitando, com isso, dificuldades de integração. Seu código interno não contém chamadas ao código legado, proporcionando todas as funcionalidades de gerência de tarefas independentes. Apesar de apresentar semelhanças com os eventos, as tarefas foram modeladas como uma classe totalmente distinta para facilitar sua inclusão ao módulo, incluindo também sua própria tabela na base de dados.

3. Conclusões

A principal contribuição deste trabalho é a apresentação de um exemplo prático de utilização de técnicas consagradas e simples de engenharia de software que permitiram uma reestruturação funcional do módulo Agenda do DiretoGNU, aliando a este capacidade de expansão e melhoria, em grande parte, de forma transparente aos usuários finais. A utilização da técnica descrita possibilitou que grande parte do código existente pudesse ser reaproveitado, diminuindo custos e tempo de projeto, além de facilitar futuras expansões e a migração do código legado.
Contudo, as modificações realizadas levaram em consideração o cenário existente. Outras alternativas poderiam ter sido utilizadas. Entretanto, sendo o DiretoGNU um produto corporativo, usado por muitas divisões do governo, sua reestruturação resultou em um balanço entre os aspectos necessidade e tempo. Uma documentação do código feita em UML (*Unified Modeling Language*) foi criada para a realização deste projeto. Os diagramas estão disponíveis [2] através de um arquivo no formato *.zargo* que pode ser visualizado através da ferramenta ArgoUML [3].

Finalmente, entende-se que este trabalho contribui também no incentivo ao uso de técnicas de Engenharia de Software no desenvolvimento de software livre, tornando-o cada vez mais inteligível e, portanto, motivando a participação da comunidade em melhorias no desenvolvimento e crescimento deste software.

Agradecimentos

Este trabalho foi financiado pela FAPERGS e pela PROCERGS.

Referências Bibliográficas

[1] Gamma, E.; et al. (1995) “Design Patterns Elements of Reusable Object-Oriented Software”. Addison-Wesley.
[2] Projeto Direto – Unilasalle. http://www.inf.lasalle.tche.br/direto, 2003
[3] ArgoUML, http://argouml.tigris.org, 2002.
[4] DiretoGNU. http://www.direto.org.br , 2003.

Transformando a Informação Pública pelo Software Livre. Um olhar sociotécnico.

Alexandre Silva Pinheiro

Coppe Sistemas – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 68511 – 21.941-972 – Rio de Janeiro – RJ – Brasil

alesilpi@cos.ufrj.br

Resumo. *Este artigo foca a adoção de software livre (e seus impactos) na informática pública através de uma abordagem sociotécnica. Usando a teoria do ator-rede para entender os diversos aspectos relacionados a este paradigma, constatamos que qualquer organização estabelecida é fruto de um amálgama de elementos humanos e não-humanos e se ousarmos modificá-la, é preciso que um número ainda maior de elementos seja mobilizado. A compreensão do comportamento desta rede de elementos heterogêneos permitirá que ações eficazes de transformação da organização sejam negociadas.*

Abstract. *This article focuses the free software adoption (and its impacts) by public information technology sectors through a sociotechnical approach. Using the Actor-Network theory to understand the various aspects related to this paradigm, we conclude that any established organization is the result from an amalgam of human and non-human elements and if we risk on change it, it's necessary that a even larger number of this elements be mobilized. The comprehension of the behavior of this network of heterogeneous elements, will allow effective organization transformation actions to be negotiated.*

1. Quando nadamos rio acima

Na história da tecnologia da informação muitos são os exemplos de proclamadas revoluções que não sobrevivem ao teste do tempo, desaparecendo completamente. Mais importante do que identificar quais as razões que levaram estas tecnologias ao fracasso é perceber que muitas delas tiveram grandes apostas de tomadores de decisão que estavam dispostos a sustentá-las[1]. As motivações pelas quais o risco é ou não assumido pelos administradores de empresas de TI por uma mudança tecnológica, pode explicar alguns dos impactos sofridos pela adoção de paradigmas novos, como o software livre.

Na grande maioria das vezes, as áreas responsáveis pela prospecção tecnológica necessitam de um lastro de confiança para que venha a adotar uma determinada solução. Consideram que arriscar seja caro (ainda que prospectar tecnologia sem risco possa parecer um contra-senso). Preferem basear-se em pareceres de terceiros[2], em uma dependência semelhante àquela submetida ao mercado financeiro pelas agências de classificação de risco. Assim, esperam seguir a tendência do mercado ao invés de analisar as soluções tecnológicas por méritos técnicos e econômicos. Deixam-se levar pela correnteza ainda que o destino seja incerto.

[1] A tecnologia PUSH foi um dos exemplos mais recentes.
[2] Instituições como o Gartner Group, IDC, entre outros.

Este comportamento também é incentivado pelas inúmeras pressões que estas áreas sofrem de fornecedores que desejam estabelecer contratos muitas vezes com o reforço de *lobbies* econômicos e políticos, especialmente a estratégia FUD (*Fear-Uncertainty-Doubt*).

A julgar por este prisma, a proposta de um modelo de produção baseado em software livre adquiri ares utópicos em uma área na qual os riscos do tratamento da informação são altos e a disposição de correr riscos é baixa. Neste cenário, como podemos motivar tomadores de decisão, pessoas e instituições para uma transformação baseada em software livre? Como nadar rio acima? Lançando um olhar sobre o processo de como estas *convicções* são construídas.

2. A informação pública: livre.

As resistências anteriormente referidas à adoção de novas soluções descrevem algumas das realidades impostas àquele que deseja executar uma mudança corporativa. Em muitos casos, talvez, a primeira ação executada é a de propor uma solução que contemple a nova tecnologia. Ela será avaliada pela área de prospecção tecnológica e após uma consulta sobre sua sustentabilidade, será aprovada ou não. Na esfera pública, o mesmo é feito com o paradigma de software livre. Fazem comparações entre soluções livres e proprietárias, mapeamentos de funcionalidades, capacitação e treinamento, suporte, custos, etc.

Essa análise, contudo, está relacionada a um *status quo* firmemente alicerçado por uma política organizacional que pretere soluções nas quais riscos devam ser assumidos ou analisados. Especialmente em relação ao software livre, a questão do suporte é repetidamente levantada como uma fraqueza da solução, ainda que possamos considerar que o suporte oferecido por fornecedores de software proprietário não signifique necessariamente que está havendo uma diminuição de riscos (apenas se paga para que um terceiro assuma a culpa pela falha) ou que efetivamente o suporte se dê pelo fornecedor (o próprio cliente resolve o problema). Outro exemplo é aquele da solução livre ignorada pela empresa que após sua inclusão em algum produto proprietário de um de seus fornecedores, passa a ser aceita como se tivesse recebido um carimbo de credibilidade. Uma espécie de intermediação entre produtor (organização livre) e consumidor (empresa pública).

A informática pública deve utilizar outros parâmetros para esta análise. O congressista peruano Edgard Villanueva Nuñez [Nuñez 2002] levantou alguns princípios básicos que justificam a adoção do software livre por governos: o acesso livre a informação pública pelo cidadão, a persistência dos dados públicos e a segurança do Estado e de seus cidadãos. Dessa forma, para que um software seja aceitável para o Estado não basta que ele seja tecnicamente capaz de cumprir uma tarefa, mas que condições contratuais futuras satisfaçam uma série de requisitos sobre sua licença, sem as quais o Estado fica impedido de garantir ao cidadão o processamento adequado de seus dados, preservando sua integridade, confidencialidade e acessibilidade. Além de atender àqueles princípios básicos, também traz conseqüências benéficas como o aumento da competitividade e o desenvolvimento do mercado regional de software[3].

Os processos de adoção de software livre geralmente são vítimas de uma abordagem reducionista do tema. É usual sua divisão em avaliações sobre a aplicabilidade em servidores, estações e ambientes de desenvolvimento e produção. Os custos dos investimentos já realizados

[3] A adoção de software livre na informática pública estimularia a competição, já que tende a gerar um suprimento de software com a melhoria do trabalho já existente, num ciclo de aperfeiçoamento contínuo, no qual a influência do marketing agressivo do mercado nas decisões do comprador é reduzida, já que a escolha passa a basear-se em méritos técnicos e não num esforço de *marketing* do fornecedor.
Com o acesso ao código-fonte, empresas poderiam prover serviços mais qualificados de suporte e não apenas aquelas transnacionais que detêm esse código fora do país.

em soluções proprietárias - o legado - e a insegurança de profissionais que temem por conseqüência do software livre, perder de prestígio (por ser especialista em uma solução proprietária), até o emprego (acreditam, de forma equivocada, que um modelo de negócio baseado em software livre não), também são freqüentemente associados à velocidade com que são implementados.

Como qualquer mudança cultural, não basta trocar-se uma plataforma pela outra. Não basta derrubar os mitos criados sobre o assunto (ainda que isto faça parte do processo). É preciso convencer, por meio da informação e de outros instrumentos, do porquê desta opção.

Contudo, estes argumentos ainda poderão ser insuficientes para que um movimento de introdução de software livre possa convencer tomadores de decisão, clientes, empregados. Por maior que seja o poder de negociação ou hierarquia utilizado[4] (ambas as características serão interessantes nesta tarefa), a mudança cultural só acontecerá quando um suficiente número de elementos humanos e não-humanos forem reunidos, mobilizados e combinados em seu favor.

3. A rede que envolve

As seções anteriores descreveram um cenário no qual argumentos a favor de mudanças tecnológicas necessitam mais que constatações técnicas, processos de negociação ou ordens hierárquicas. Neste quadro, mesmo que evidências técnicas ou políticas sejam claras a favor de soluções livres, o efetivo engajamento da organização ainda dependerá do *interesse* em abandonar a realidade atual por uma outra.

Aprofundando um pouco a análise, olhando sobre os ombros da tecnociência, percebe-se que indivíduos e *coisas* sustentam afirmações, crenças e interesses que *defendem* e *constituem* a realidade tecnológica vigente. Se olharmos *mais de perto*, os analistas da área tecnológica da empresa que resistem, as agências de ranqueamento, os fornecedores, os lobbies, os sistemas de legado, a falta de uma política corporativa sobre o tema, etc., fazem parte de uma lista de elementos heterogêneos que estão ligados entre si formando uma *rede* [Latour 1987] pelo qual se define e conforma a *realidade*.

Quando estamos frente a essa rede, três caminhos são possíveis: ser absorvido por ela, ser ignorado ou confrontá-la. Nas três opções, ocorre uma *acusação de irracionalidade* por não fazermos parte dela. A *controvérsia* gerada pelo discordante então permite que a forma como as atribuições de causa e efeito foram realizadas, de como estes elementos estão interligados e quão forte são essas ligações sejam observadas na rede. A razão (ou sua ausência), portanto, é um resultado não de seus atributos intrínsecos, mas de uma *sócio-lógica* na qual os resultados dependem de *provas de força* sobre as associações entre os elementos da rede.

> **"A acusação de irracionalidade é sempre feita por alguém que está construindo uma rede sobre outra pessoa que se mantém no seu caminho; portanto, não há linha divisória entre mentes, mas apenas redes maiores ou menores". [Latour 1987]**

Atuar na modificação desta realidade temporária, de algo que naturalizado explica-se por si mesmo, de um *fato*, significa a criação de uma outra (realidade, rede) que sirva ao objetivo da mudança pretendida. Esta somente acontecerá após a expansão do domínio desta segunda.

Portanto, a construção de uma rede que difunda a afirmação "*software livre deve ser usado por empresas públicas"*, exigirá uma amarração de um número ainda maior de elementos heterogêneos: artigos, aliados técnicos, aliados políticos, inscrições, caixas-pretas,

[4] Sobre autoridade através de hierarquia ou de negociação ver Negotiating Corporate Change da Harvard Business School, por James K. Sebenius.

tradução/traição de interesses, etc, representando um aumento do custo de contestar-se a realidade que, por enquanto, vigora. A dinâmica do *ator-rede* que mobilizará estes esforços e a evolução da rede dependerá desta combinação de associações de recursos humanos e não-humanos dentro de *laboratórios*[5] onde estes elementos são reunidos.

"A abordagem do ator-rede é a assim uma teoria de agentes, uma teoria de conhecimento, e uma teoria de máquinas. E, mais importante, ela diz que nós devemos explorar os efeitos sociais, seja qualquer sua forma material, se quisermos responder o "como" de questões sobre estrutura, poder e organização." [Law 1992] (tradução)

O sucesso (ou fracasso) da rede dependerá de como estes elementos serão recrutados, combinados e devolvidos em seu movimento de expansão. Este trabalho deve ser encarado pelo construtor de fatos como um *empreendimento* [Law 1989].

Conclusão

Apresentamos uma proposta de analisar a adoção de software livre não mais sobre uma ótica reducionista na qual o processo é encarado apenas como uma substituição tecnológica, sem levar em consideração os efeitos sociotécnicos que são causa e conseqüência de seu sucesso ou fracasso.

O arcabouço dessa análise será a teoria do ator-rede, pretendendo que a discussão acerca do tema salte dessa visão reducionista comumente utilizada para novos patamares de compreensão e, partindo deste ponto, através de um enfoque simétrico sobre estudos de casos, indicar modelos e estratégias que atendam às necessidades para sua efetivação.

Finalmente, uma ação coordenada que permita a adoção de software livre nas diversas esferas governamentais como visão de futuro, minorando o desgaste de ações isoladas (não colaborativas) e/ou duplicadas (re-trabalho) e estabelecendo um modelo que seja aderente a todas.

Referências bibliográficas

Callon, M., (1989) "A agonia de um laboratório", in Callon, M., "La science et sés reseau", tradução de Ana Lúcia Villasboas, revisão de Ivan da Costa Marques, Paris, La Découverte.

Latour, B. (1987) "Ciência em ação: como seguir cientistas e engenheiros sociedade afora", tradução de Ivone C. Benedetti, revisão de Jesus Paula de Assis, Editora UNESP, São Paulo, Brasil, 2000.

Latour, B., (1989) "Pasteur e Pouchet: Heterogênese da história das ciências", in: Serres, M. (org.), Elementos para uma História das Ciências III. De Pasteur ao Computador, Terramar, Lisboa, 1996.

Law, J., (1989) "O laboratório e suas redes", in Callon, M., "La science et sés reseau", tradução de Ana Lúcia Villasboas, revisão de Ivan da Costa Marques, Paris, La Découverte.

Law, J., (1992) "John Law' Notes on the Theory of the Actor Network: Ordering, Strategy an Heterogeneity", Centre for Science Studies, Lancaster University, Lancaster, UK.

Nuñez, E. (2002) "Carta do congressista em resposta ao Gerente Geral da Microsoft / Peru", http://www.gnu.org.pe/resmseng.html

[5] Para um estudo do comportamento dos atores-rede e seus laboratórios, ver [Callon 1989].

Estudo Comparativo Entre Ferramentas Gratuitas Versus Proprietárias em um Processo de Desenvolvimento de Software Orientado a Objetos

Eduardo Quesado Filgueiras[1, 2], Fábio Gomes Naveca[1]

[1]Centro Universitário Luterano de Manaus (CEULM-ULBRA)
Av. Solimões, 02, Conj. Atílio Andreazza – Manaus – AM

[2]Centro Regional de Vigilância de Manaus – Sistema de Proteção da Amazônia (SIPAM)
Av. Turismo, 1350, Tarumã – Manaus – AM

quesado@ulbra-mao.br, fabio@it.atech-mn.br

Abstract. *This paper describes the results of a case study performed between fully free and proprietary software development platforms, including operating systems, CASE tools, integrated development environments, database management systems, project management and version control tools. Positive and negative characteristics associated to each environment are described and evaluated, using Software Engineering metrics.*

Resumo. *Este artigo descreve os resultados de um estudo de caso realizado entre ambientes de desenvolvimento puramente baseados em software livre, em oposição a ambientes semelhantes de código proprietário, incluido sistemas operacionais, ferramentas CASE, ambientes integrados de desenvolvimento de software, sistemas gerenciadores de bancos de dados e ferramentas de gerenciamento de projeto e controle de versão. As características positivas e negativas associadas a cada ambiente são descritas e avaliadas utilizando Métricas de Engenharia de Software.*

1. Introdução

O campo da Engenharia de Software é uma das áreas cujo gradiente evolutivo é dos mais elevados na Ciência da Computação, devido a grande variedade de situações apresentadas pela dinâmica dos usuários. A essência da Engenharia de Software, no entanto, concentra-se nas ferramentas, métodos e procedimentos utilizados no processo de desenvolvimento [Pressman 2001]. Neste contexto, tem sido ampla a discussão sobre uma nova filosofia que visa a alteração radical do paradigma de desenvolvimento de software, o projeto GNU e o software livre. "Software livre se refere à liberdade dos usuários executarem, copiarem, distribuírem, estudarem, modificarem e aperfeiçoarem o software" [Williams 2002]. Não existe hoje, porém, um processo objetivo para saber se ferramentas de software livre no contexto de desenvolvimento de software dispõem de recursos e desempenho semelhantes a ferramentas já consagradas do mercado, senão por um estudo comparativo entre duas aplicações que enfoquem o mesmo domínio do problema, onde sejam definidos parâmetros comparativos a fim de que se verifiquem as vantagens e as desvantagens de cada uma das plataformas, em seu conjunto [Filgueiras 2003]. Dessa forma, diferentes tecnologias associadas ao desenvolvimento de software foram escolhidas para a implementação do estudo, como gerenciadores de projeto, de banco de dados (SGDB), e de versão de software, ferramentas de engenharia de software auxiliada por computador (CASE), além de sistemas operacionais, compiladores e ambientes integrados de desenvolvimento, considerando-se, porém, a necessária consistência entre os elementos a serem comparados, em termos de funcionalidade, bem como a definição das métricas apropriadas e aplicáveis ao processo.

2. Processo Comparativo

Em se tratando de processo de desenvolvimento de software, o processo comparativo a ser realizado entre diferentes ambientes se torna mais complexo devido ao grande número de fatores que estão envolvidos na elaboração de um projeto de software. Sendo assim, as primeiras iniciativas necessárias foram a validação da consistência de meios e a determinação das possíveis métricas associadas.

Segundo Naveca e Filgueiras (2002), "para que seja feita uma comparação entre plataformas de desenvolvimento de software de forma apropriada é necessário, antes de tudo, definir algumas restrições a serem seguidas, a fim de que o processo de comparação seja cientificamente consistente, expurgando-se as variáveis indesejáveis à consistência da comparação**".** Dessa forma, a fim de manter a coerência entre ambientes de desenvolvimento e evitar polarizações indesejadas no processo comparativo, foram escolhidos o paradigma orientado a objetos [Booch 1994], a Linguagem de Modelagem Unificada (UML) [Booch et al 1998] e a linguagem de programação Java [Eckel 2000] como motores do processo, pois estes são independentes de plataforma de desenvolvimento, o que confere portabilidade e flexibilidade ao modelo comparativo.

Neste contexto, as ferramentas escolhidas, consideradas como elementos mínimos para o desenvolvimento de software com qualidade estão listadas na Tabela 1.

Tabela 1 - Tecnologias e Ferramentas de Desenvolvimento de Software

Tecnologia	Ambiente Proprietário	Ambiente de Software Livre
Sistema Operacional	MS Windows 2000	RedHat Linux 7.3
Ambiente de desenvolvimento	Oracle JDeveloper 9i	Netbeans 3.2
Ferramenta CASE	Rational Rose 2000	Argo UML 0.10
Controle de versão	MS Visual SourceSafe 6.0	CVS 1.11.2
Sistema Gerenciador de Banco de Dados	Oracle 9i	PostgreSQL 7.2.2
Documentação de Projeto*	MS Office 2000	OpenOffice 1.0
Planejamento*	MS Project 2000	MrProject 0.9

*Resultados não apresentados neste trabalho

3. Escolha das Métricas Apropriadas

A medição é um fator comum ao mundo da engenharia, já que permite a definição de parâmetros para escolhas cientificamente embasadas. No caso da Engenharia de Software, entretanto, há uma dificuldade adicional em se concordar sobre o que deve ou não ser medido. As medições do mundo físico podem ser divididas em duas categorias: *diretas* e *indiretas* [Pressman 2001].

Neste estudo, o contexto associado foi a exploração das métricas indiretas, isto por que não se está preocupado em verificar a quantidade de linhas existentes em um código, mas sim a qualidade com que essas linhas de código foram escritas e o nível de produtividade dos técnicos que elaboram esse produto. Baseando-se nisto, realizou-se um trabalho com o intuito de definir métricas para realização de um processo de desenvolvimento de software com qualidade, chegando-se ao seguinte conjunto: correção, confiabilidade, eficiência, integridade, usabilidade, manutenibilidade, flexibilidade, testabilidade, portabilidade, reusabilidade, interoperabilidade, escalabilidade, segurança do código e do modelo, autodocumentação, automatização, treinamento, continuidade e estabilidade [Naveca e Filgueiras 2002].

4. Resultados

É importante ressaltar que, para cada par de ferramentas de código proprietário e aberto, foram definidos subconjuntos das métricas citadas que se adequassem ao contexto individual sem perda de consistência e significado. Para que o resultado do estudo obtivesse um melhor entendimento, foram atribuídos graus de valores **1**, **3** e **5** para os desempenhos relativos a cada ferramenta, ao se considerar software de funcionalidades e características semelhantes, significando, respectivamente, "RUIM", "MEDIANO" ou "BOM". Desse modo, pôde-se construir a Tabela 2 contendo os resultados de comparação para cada tipo de ferramenta associada ao estudo.

Uma avaliação importante, com relação ao sistema operacional, é o evidente crescimento do sistema operacional Linux, por sua confiabilidade, robustez e estabilidade, em relação ao sistema operacional Windows. Porém, não se pode negar que a facilidade de uso do último, bem como de todo software desenvolvido para a plataforma, constitui um atrativo muito forte para o usuário. Ao se tratar da ferramenta de programação, o JDeveloper leva vantagem frente ao Netbeans com relação ao desempenho do programa. Outrossim, as duas ferramentas estão evoluindo e englobando tarefas antes realizadas separadamente por outras ferramentas, como modelagem e controle de versão. Em se tratando de ferramentas que auxiliam desenvolvimento utilizando UML (CASE Tools), as vantagens do Rational Rose frente ao ArgoUML são alusivas à quantidade de recursos, como o maior número de linguagens de programação suportadas para geração de código, além da capacidade de engenharia reversa, bem como suporte a estereótipos de análise robusta. O software livre implementa, entretanto, modelos seguindo fielmente a especificação UML e gera código condizente com as classes implementadas. Quanto ao controle de versão, a formalidade inerente à utilização de ferramentas deste tipo e a complexidade de preparação e gerenciamento do repositório de código para utilização em ambiente multiusuário evidenciam que tanto o ambiente livre quanto o proprietário carecem de amigabilidade. Com relação ao gerenciador do banco de dados, a fácil integração do SGBD Oracle e a ferramenta de desenvolvimento já eram evidentes, bem como a conhecida escalabilidade. O banco de dados PostgreSQL, outrosim, é excelente implementação de SQL padrão, possuindo inclusive extensões objeto-relacional e espacial.

5. Conclusão

A comparação entre ambientes de programação de paradigmas distintos se mostra necessária e útil para mostrar não somente uma nova forma de implementação de sistemas que adota a abertura de código-fonte, mas também para apresentar como o novo paradigma vem conquistando mercados dominados anteriormente por código proprietário, agregando mais adeptos, de forma cada vez menos passional e mais científica. O trabalho pôde mostrar diferentes tecnologias em oposição a sistemas conhecidos no meio comercial e tidos como padrão de desenvolvimento de software. Finalmente, observa-se que hoje o software proprietário tem grande aceitação devido aos recursos disponibilizados por interfaces de fácil utilização. O software livre, porém, realiza as mesmas funcionalidades que o proprietário, com menor custo inicial, o que permite a extrapolação de um ambiente futuro em que mais e mais usuários utilizarão gradativamente o software de código aberto, provocando um crescimento no número de desenvolvedores para atender a demanda, o que levará a uma reação em cadeia com aumento ainda maior dos usuários, realimentando o processo em uma espiral ascendente. Em resumo, o software livre tende a crescer mais e mais. Importantes discussões futuras, entretanto, são a continuidade do software livre e os aspectos econômico-financeiros, avaliando-se os custos agregados relativos a um ambiente totalmente gratuito e outro tradicional, no tocante ao custo total de propriedade (TCO).

Tabela 2 - Resultados comparativos

Ferramenta	Métrica	Livre	Proprietário	Observações
Sistema operacional	Confiabilidade	5	1	É perceptível que o sistema operacional Linux já supera o Windows em vários aspectos, evidenciando o gradiente evolutivo positivo do primeiro. Necessita, entretanto, continuar investindo em interface com o usuário.
	Integridade	5	3	
	Eficiência	3	3	
	Usabilidade	1	5	
	Estabilidade	5	1	
	Total	**19**	**13**	
Ambiente de desenvolvimento	Continuidade	5	5	O software livre possui as mesmas características relevantes essenciais, mas é necessário um melhor desempenho para que tenha a mesma aceitação do código proprietário.
	Autodocumentação	5	5	
	Reusabilidade	5	5	
	Usabilidade	1	3	
	Eficiência	1	3	
	Total	**17**	**21**	
Ferramenta de modelagem	Automatização	5	5	O Rational Rose é mais rico em características assessórias, como geração de código não Java e integração com o SGBD, porém o ArgoUML implenta integralmente a especificação UML.
	Segurança do modelo	5	5	
	Usabilidade	5	5	
	Eficiência	3	5	
	Corretitude	5	5	
	Total	**23**	**25**	
Controle de versão	Segurança de código	5	5	Com relação as tarefas nativas esperadas de ferramenta de controle de versão, o CVS obteve desempenho idêntico ao SourceSafe, mas é necessária uma interface mais amigável e intuitiva.
	Usabilidade	1	3	
	Confiabilidade	5	5	
	Integridade	5	5	
	Treinamento	1	3	
	Total	**17**	**21**	
SGBD	Confiabilidade	5	5	O SGBD proprietário é superior em quase todos os aspectos, porém o PostgreSQL tem evoluido em características e *performance* e certamente já superaria outros concorrentes no mercado que não o Oracle.
	Integridade	3	3	
	Escalabilidade	3	5	
	Interoperabilidade	3	5	
	Estabilidade	3	5	
	Total	**17**	**23**	
TOTAL		**93**	**102**	Vantagem do ambiente proprietário sobre o livre não é significativa.

Referências

Booch, G. “Object-Oriented Analisys and Design With Applications", Addison-Wesley, 1994.

Booch, G., Rumbaugh, J., Jacobson, I. “The Unified Modeling Language User Guide”, Addison-Wesley, 1998.

Eckel, B. “Thinking in Java”, Prenticce Hall, 2000.

Filgueiras, E.Q. . “Migrando de um Ambiente Proprietário para Software Livre”, Developers’ Magazine, n° 80, Axcel Books, p. 16-19. Abril de 2003.

Naveca, F. G. e Filgueiras, E. Q. “Software Livre: Métricas para um Estudo Comparativo”, Developers’ Magazine, n° 72, Axcel Books, p. 14-16. Agosto de 2002.

Pressman, R. S. “Software Engineering: a Practitioner’s Approach”, McGraw-Hill, 2001.

Williams, S. “Free As In Freedom: Richard Stallman´s Crusade for free software”, O´Reilly & Associates, 2002.

Um Modelo Espacial para Integração dos Dados da Saúde Implementado em Software Livre

Vania Bogorny[1], Felipe Nievinski[2], Nara Bigolin[3]

[1] *Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS) - Campus do Vale, Av. Bento Gonçalves, 9500, Bloco 4, Bairro Agronomia, Porto Alegre, RS, CEP: 91.501-970, Brasil*

[2] Instituto de Geociências - Universidade Federal do Rio Grande do Sul (UFRGS) - Campus do Vale, Av. Bento Gonçalves, 9500, Bloco 4, Bairro Agronomia, Porto Alegre, RS, CEP: 91.501-970, Brasil

[3] Faculdade de Informática – (ULBRA) - Rua Miguel Tostes, 101 – Canoas – RS -Brasil
vania@inf.ufrgs.br, fnievinski@terra.com.br, nara@ulbra.tche.br

Resumo*: o software com o código aberto está sendo desenvolvido com agilidade e está ganhando mercado em várias áreas da informática, incluindo sistemas operacionais, bancos de dados convencionais e geográficos, linguagens de programação, sistemas de descoberta de conhecimento, entre outros. Este artigo apresenta um modelo de dados, orientado a objetos, que integra e referencia, geograficamente, as bases de dados heterogêneas da Secretaria Estadual de Saúde do Estado do Rio Grande do Sul (SES/RS), que foi implementado em software livre.*

Abstract: *open source software is being increasingly developed in a number of Information Technology areas, including operating systems, traditional database management systems, spatial DBMS, programming languages, knowledge discovery systems, among many others. This paper presents an object data model that integrates and spatially represents the heterogeneous databases related to public health of Rio Grande do Su, implemented in free software tools.*

1. Introdução

Os Sistemas de Informação Geográfica (SIG) são sistemas especiais que tratam dados convencionais/descritivos e a parte espacial dos dados geográficos, dados estes que têm uma localização com relação a superfície terrestre [Câmara 1996]. Os dados geográficos estão sendo cada vez mais utilizados por empresas privadas e órgãos governamentais para gerenciar questões ambientais, redes de telefonia e energia elétrica, áreas urbanas, epidemias, questões de saúde pública, entre outras. A Coordenadoria de Política de Atenção Integral a Saúde do Trabalhador da SES/RS optou por um SIG para fazer o mapeamento de intoxicações, internações e mortes decorrentes de acidentes e doenças com uma possível causa relacionada ao ambiente de trabalho. Além disso, o sistema permite, também, o mapeamento de acidentes e doenças não relacionadas ao local de trabalho, podendo assim, ser utilizado em diversas áreas de saúde pública.

O modelo de dados do SIG, denominado GEOSIST (Georreferenciamento do Sistema de Informações em Saúde do Trabalhador) integra as seguintes bases de dados: Sistema de Informações de Mortalidade (SIM), Sistema de Informações Hospitalares (SIH), Sistema de Informações de Notificação de Agravos Notificáveis(SINAN) e o Sistema de Informações em Saúde do Trabalhador (SIST). Através de sua interface WEB [Bogorny, Nievinski, Bigolin, Kalil e Corrêa 2002], o usuário pode fazer consultas e análises espaciais cruzando dados de agravo (acidentes e doenças) e dados de risco (possíveis causas que ocasionaram o acidente ou a doença). Os últimos foram coletados de órgãos externos à SES como o IBGE (Instituto Brasileiro de Geografia e Estatística), o Ministério do Trabalho, a Secretaria de Agricultura e Abastecimento (SAA) e o Programa Pró-Guaíba. A análise dos

dados pode ser realizada mapeando o Estado do RS como um todo ou mapeando as coordenadorias regionais de saúde.

Este artigo apresenta um modelo de dados que integra e espacializa as diferentes bases de dados SES/RS, especifica algumas dificuldades encontradas na integração dos dados e menciona os benefícios por ter sido implementado na plataforma livre. O restante do artigo está organizado da seguinte forma: a Seção 2 apresenta uma visão geral do modelo de dados de Agravos. A Seção 3 aborda algumas dificuldades encontradas no levantamento e na integração dos dados. A Seção 4 cita os benefícios do modelo por ter sido implementado na plataforma livre. A Seção 5, por sua vez, apresenta as conclusões e os trabalhos futuros.

2. Modelo de Integração dos Dados

A base de dados descritivos do GEOSIT foi criada a partir de dois modelos: um modelo de *agravos* e um modelo de *riscos*. Ambos foram baseados no formalismo da orientação a objetos, usando a notação UML [Booch 2000]. A figura 1 ilustra o diagrama de classes dos agravos.

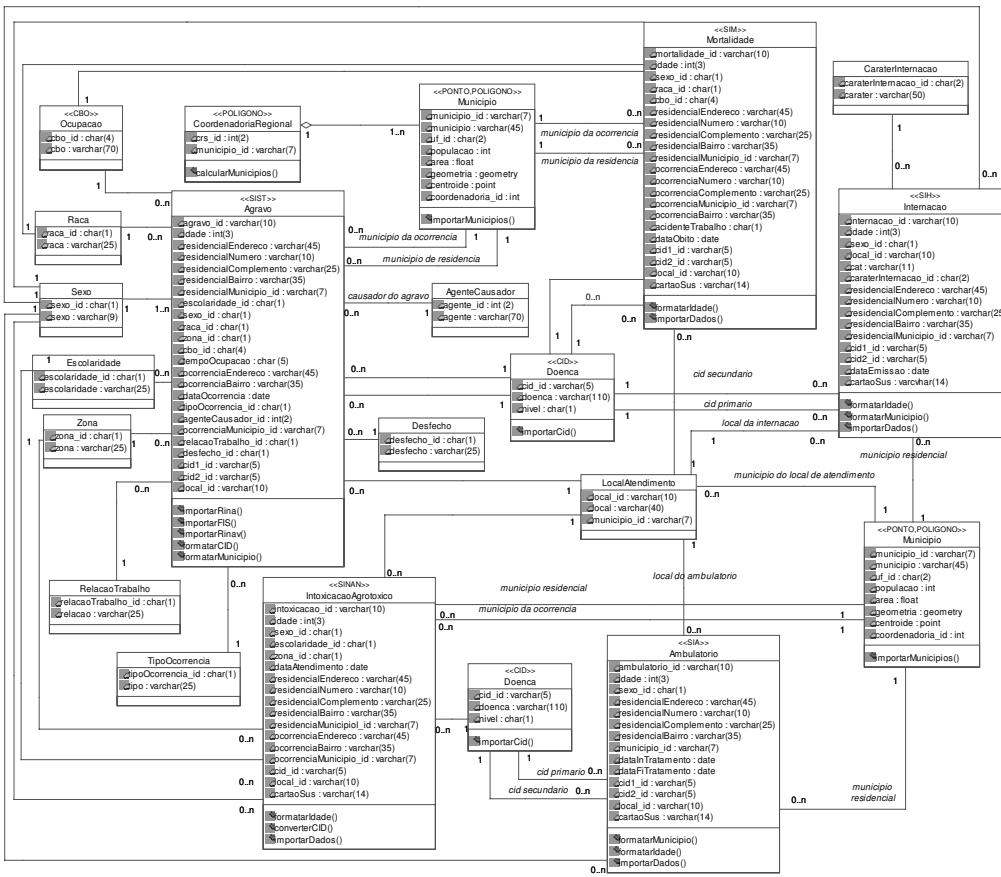


FIGURA 1 – Diagrama de classes dos agravos

No diagrama de classes da figura 1, estão representados os *agravos* mais relevantes para a Coordenadoria de Política de Atenção Integral à Saúde do Trabalhador, entre os quais estão as intoxicações causadas por agrotóxicos, as internações hospitalares, as mortes, os acidentes e os casos de violência. O estereótipo da classe expressa o sistema de informação que os originou os objetos.

Os dados de *risco* foram modelados com base em entrevistas realizadas com usuários de órgãos externos à SES/RS. Entre os principais elementos desse modelo estão os acidentes de cargas perigosas, a distribuição agrária, os pontos de coleta de água, as principais fontes poluidoras, os pontos de lançamento de líquido e a medição da poluição atmosférica. O modelo completo de riscos está descrito em [Bogorny e Nievinski 2002a].

A base de dados geográfica é formada por três mapas distintos, contudo, apresentados na mesma escala, permitindo a realização de camadas de sobreposição. Esses mapas são os Limites Políticos Municipais, que incluem os atuais 499 municípios gaúchos (fonte –SAA), o mapa Hidrográfico e o mapa Rodoviário estaduais. O mapa de Limites Políticos Municipais é a base para a espacialização e o mapeamento temático dos dados de atributos.

3. Dificuldades Encontradas na Realização do Projeto

O GEOSIST engloba dados de múltiplas fontes que, por conseqüência, são em grande parte heterogêneos. As diferentes fontes de dados utilizam codificações e padrões próprios, dificultando a sua integração. Em virtude do tempo previsto para o término do projeto, um modelo de dados foi criado para integrar as diferentes bases de dados, porém, apenas os dados de município e doenças foram integrados para fazer as consultas e análises espaciais. Os municípios foram codificados de acordo com os padrões estabelecidos pelo IBGE e as doenças de acordo com o Código Internacional de Doenças (CID) [Bogorny, Nievinski, Bigolin, Kalil e Corrêa. 2002].

Um fator importante que influencia diretamente nas consultas e análises espaciais é a quantidade de dados com ruído, ou seja, dados nulos e dados preenchidos inadequadamente. Outro aspecto relevante é o número de informações coletadas e tratadas de forma manual, ou seja, que não estão disponíveis em formato digital. Outros dados de fundamental importância para o sistema como, por exemplo, a quantidade de agrotóxico fabricado/importado e consumido nos diferentes municípios do estado, não são controlados pela SAA.

Além das dificuldades mencionadas, não existe, ainda, um consenso entre empresas e órgãos governamentais para padronizar, disponibilizar e compartilhar informações. Isso dificulta a total implementação de projetos como o GEOSIST. Em virtude desse tipo de problema, muitos dados, principalmente os de risco, foram mapeados na fase de concepção do sistema, criados no projeto lógico, porém não instanciados no projeto físico.

4. Benefícios/Facilidades propiciadas pelo uso do software livre

O GEOSIST foi totalmente desenvolvido em software livre. Sua arquitetura [Bogorny, Nievinski, Bigolin, Kalil e Corrêa 2002] é formada de uma interface WEB desenvolvida em *PHP* [PHP 2003], através da qual o usuário acessa a base de dados descritiva e espacial que é gerenciada pelo *PosgreSQL* [PostgreSQL 2003] e *PostGIS* [PostGIS 2003]. Os mapas são gerados pelo *UMN-MapServer* [MapServer 2003].

Os principais benefícios da utilização de software livre para este trabalho foram:

- *existência e disponibilidade dos recursos de software*: foram encontrados programas adequados para a implementação de cada etapa do projeto;
- *otimização de consultas*: o acesso ao código fonte permitiu adaptar os programas de acordo com as necessidades do projeto. Mais especificamente, a conexão do *UMN-MapServer* com o *PostgreSQL* foi adaptada para acelerar a execução de consultas complexas sobre os dados espaciais. O acesso ao

código fonte também permitiu a correção de dois *bugs* pequenos no *UMN-MapServer*, cujas correções foram incorporadas à distribuição oficial desse software;

- *documentação*: apesar de não haver uma empresa responsável pelo suporte, como existe para softwares proprietários, os principais programas utilizados disponibilizam uma ótima documentação e oferecem acervos como listas de discussão que facilitam a resolução de diversos problemas.

5. Conclusões e trabalhos futuros

Apesar das dificuldades encontradas na busca pela informação, no mapeamento e na padronização dos dados, o GEOSIST pode auxiliar o especialista em saúde pública a tomar medidas preventivas com relação às mortes, internações hospitalares e acidentes e doenças possivelmente relacionadas ao ambiente de trabalho.

O modelo de dados, apresentado neste artigo foi utilizado para fazer o mapeamento espacial dos agravos em saúde pública. Ele pode servir de base para integrar os diferentes sistemas de informação existentes na SES, desde que os dados como sexo, raça/cor, escolaridade, entre outros, sejam padronizados. O mapeamento desses dados está descrito em [Bogorny e Nievinski 2002a].

Com relação aos trabalhos futuros, inúmeras melhorias podem ser realizadas para que o GEOSIST se torne um SIG completo para o cruzamento espacial de agravos e riscos. Inicialmente é necessário padronizar dados como faixa etária, sexo, raça, escolaridade, ocupação, etc. Um outro aspecto de fundamental importância é a importação de todos os dados de risco, que foram modelados, porém, não foram disponibilizados pelos órgãos responsáveis.

Dados como uso de agrotóxicos, quantidade de água e percentual de esgoto tratado, classe social, entre outros, são informações que, de uma maneira geral, aumentariam a qualidade do trabalho.

Os dados do GEOSIST, principalmente os que se referem às mortes e internações estão sendo utilizadas como fontes de pesquisa na área de Inteligência Artificial, mais especificamente na Descoberta de Conhecimento em Bases de Dados (DCBD) [Lucas, Bigolin e Alvares 2002]. Nesta área, também estão sendo utilizadas ferramentas de software livre como, por exemplo, o Weka [Weka 2003] . Futuramente o GEOSIST deve ser utilizado como base para a Descoberta de Conhecimento em Bases de Dados Geográficos.

6. Referências Bibliográficas

Bogorny, V. Nievinski, F.; Bigolin, N.; Kalil, F.; Corrêa, M.J.. (2002), Desenvolvimento de um SIG para Saúde Pública usando Software Livre. Workshop de Software Livre.

Bogorny, V.; Nievinski, F. (2002), Projeto Conceitual, incluindo análise de requisitos e levantamento dos dados e dos softwares livres para o projeto de SIG. Relatório Técnico.

Bogorny, V.; Nievinski, F. (2002a), Criação Física do Banco de Dados. Relatório Técnico.

Booch, G.; Rumbaugh, J.; Jacobson, I. (2000), UML: user guide. Rio de Janeiro: Campus.

Câmara, G.; Medeiros, C.B.; Casanova, M.A.; Hemerly, A.; Magalhães, G. (1996) "Anatomia de Sistemas de Informação Geográfica", Instituto de Computação, UNICAMP – Campinas.

Lucas, A. M.; Bigolin, N. M.; Alvares, L. O. Mineração em dados reais da saúde considerando aspectos temporais. Uruguaiana-RS, p. 97-104, 2002.

PHP (2003), http://www.php.net/.

PostgreSQL (2003), http://www.postgresql.org.

PostGIS (2003), http://postgis.refractions.net.

UMN-MapServer (2003), http://mapserver.gis.umn.edu.

Weka 3: Machine Learning Software in Java (2003), http://www.cs.waikato.ac.nz/ml/weka.

Estudo do Processo de Migração de Software Proprietário para Software Livre para Aplicações Web

Patrícia Sartor, Alexandre Moretto Ribeiro

Universidade de Caxias do Sul
Rua Francisco Getúlio Vargas, 1130 – 95.070-560 – Caxias do Sul – RS – Brasil
psartor@ucs.br, aribeiro@ucs.br

Abstract. *This paper presents a study about the process of migrating applications from Web Proprietary Software to Web Free Software. The collaborative development model was studied for implementing it and the main free Web development tools available. Besides, in order to evaluate a real case of migration and the tools used, the process of migrating the RECRIA wich is a social service's family registration system is described.*

Resumo. *Neste trabalho é apresentado um estudo do processo de migração de aplicações Web construídas com Software Proprietário para aplicações Web construídas com Software L*ivre*. Foi estudado o modelo de desenvolvimento colaborativo e as principais ferramentas* livres *de desenvolvimento Web existentes no mercado. Para possibilitar a avaliação em um caso real do processo e das ferramentas estudadas, foi desenvolvido um estudo de caso envolvendo o cadastro de famílias da RECRIA.*

Introdução

A Prefeitura de Caxias do Sul, através da Fundação de Assistência Social (FAS), responsável pela gestão da assistência social no município, faz parte da Rede de Atenção à Criança e ao Adolescente (RECRIA), que é um grupo de entidades governamentais, não-governamentais, instituições de justiça e conselhos setoriais que trabalham no atendimento de crianças, adolescentes e suas famílias. A FAS, através de recursos do Banco Nacional de Desenvolvimento Econômico e Social (BNDES) e da Fundação Telefônica, implementou um Sistema de Informações para integrar as entidades da RECRIA de forma informatizada. Foram criados um site e um aplicativo com um banco de dados que pode ser acessado através desse site [Recria 2001]. Esse aplicativo consiste de um cadastro único de famílias atendidas pelas entidades e de um módulo que implementa os trâmites dos adolescentes em conflito com a Lei. Ao todo, são quase cinqüenta programas assistenciais do município que acessam este sistema.

Quando foi iniciada a implementação do aplicativo para a Rede de Atenção à Criança e ao Adolescente de Caxias do Sul, não foi levada em consideração a possibilidade de utilizar ferramentas de *software* livre para a sua construção. O aplicativo foi concebido utilizando *software* proprietário: o banco de dados utilizado é o Microsoft SQL Server 2000 [Ramalho 2001]; o ambiente de programação é baseado na metodologia Windows DNA [Otey and Conte 2001] e utiliza ASP [Homer 2000] (para as páginas Web) e Visual Basic (para os componentes). O aplicativo está fisicamente em um servidor, com sistema operacional Windows 2000, dentro da Divisão de Informações da Prefeitura de Caxias do Sul. Este é um dos únicos servidores com sistema operacional proprietário dentro da Divisão de Informações da Prefeitura, sendo que, atualmente, o sistema corporativo da Prefeitura está sendo migrado para a linguagem Java, o que possibilitará, em breve, uma total migração para sistemas operacionais livres, ficando apenas o aplicativo da RECRIA sob plataforma proprietária.

O Uso do *Software* Livre na Área Social

Pelo Brasil, existem outras cidades que também possuem um trabalho semelhante ao da RECRIA, ou estão iniciando sua construção, sempre com financiamento de bancos, como é o caso do BNDES, ou de fundações que repassam recursos de empresas. Porém, o envolvimento dos bancos e das fundações não fica apenas na área financeira. Eles são parceiros no encontro de soluções para os problemas de cada município, monitoram e avaliam com muito cuidado os resultados dos investimentos. Por isso, perceberam que uma das grandes dificuldades dos municípios era a informatização da Rede. Por não existir um modelo a seguir e também pelo desconhecimento de informática dos trabalhadores da área social, muitos dos recursos foram disperdiçados e/ou mal empregados em sistemas que não atendem as necessidades dos usuários.

Pelos problemas acima descritos, percebe-se que a solução é desenvolver um aplicativo para gerenciamento das Redes de Atenção que seja construído com *software* livre e que ele próprio seja um *software* livre. Isso facilitaria a disseminação do sistema para vários municípios e sua manutenção sem altos custos, já que, por ter seu código aberto, qualquer pessoa com conhecimento das linguagens de programação empregadas para sua construção poderia modificá-lo. Além disso, seria um sistema sempre em crescimento, já que todos as municípios interessados poderiam colaborar com sua construção e enriquecê-lo conforme suas experiências locais.

Migração de Aplicações Web Proprietárias para Aplicações Web Livres

Os projetos de *software* livre atuais caracterizam-se pelo grande número de usuários e desenvolvedores, e pela discussão pública das decisões de desenvolvimento e implementação.

O modelo de desenvolvimento colaborativo de *software* livre foi descrito por Eric Raymond em seu livro "The Cathedral and the Bazaar" [Raymond 1999]. Raymond chama o desenvolvimento de *software* proprietário de "Catedral", no qual um grupo pequeno de arquitetos trabalham em relativo isolamento para criar um produto grande, complexo e valioso.

Já a forma de implementação de *software* livre é como um bazar, no qual inúmeras pessoas participam do processo de construção de *software*, trocando informações entre si num ritmo quase caótico [Reis 2001]. É claro, porém, que não é possível desenvolver *software* livre desde o início no estilo bazar. Quando um *software* é apresentado a uma comunidade, ele pode ser "grosseiro, cheio de erros, incompleto e pobremente documentado", porém, é imprescindível que seja executável e passível de testes. Deve também convencer o usuário (co-desenvolvedores em potencial) de que ele pode evoluir para um *software* elegante em breve.

Com o propósito de verificar a possibilidade de migrar aplicativos web construídos com *software* proprietário para plataformas livres, foi realizado o estudo de algumas ferramentas proprietárias e livres. As ferramentas proprietárias estudadas foram: ASP, Visual Basic (Arquitetura Windows DNA), MS SQL Server 2000 e Oracle [Die 1999]; as ferramentas livres estudadas foram: PHP [PHP 2001], Zope [Zope 2001], MySQL [MySQL 2001] e PostgreSQL [PostgreSQL 2002].

A migração de um aplicativo *Web* escrito com *software* proprietário para um aplicativo desenvolvido com ferramentas livres pode ser dividida em três etapas. Em cada etapa será migrada uma das camadas da aplicação: camada de interface com o usuário, camada de aplicação ou de negócios e camada de acesso ao banco de dados.

Camada de Interface com o Usuário

A interface de um aplicativo *Web* desenvolvido com ferramentas proprietárias pode ser totalmente reutilizada, quando escrita em HTML. O código fonte da página HTML pode ser visualizado através de qualquer editor e as modificações necessárias podem ser feitas. Via de regra, não é necessário modificar o código, apenas retirar as partes de código que não são

HTML, tais como chamadas a componentes escritas em ASP ou até mesmo códigos *Javascript* ou *Vbscript* que possam estar embutidos em meio ao HTML. Os códigos ASP e outros *scripts* que foram retirados devem ser substituídos por seus correspondentes no mundo do *software* livre.

Aplicativos escritos utilizando a metodologia Windows DNA possuem partes de código ASP escrito em meio ao código HTML. Esses códigos ASP nada mais são que chamadas para componentes da camada de negócios. Numa possível migração para uma plataforma livre, o código HTML pode ser totalmente reutilizado, tanto utilizando PHP quanto Zope.

Camada de Aplicação ou de Negócios

A camada de aplicação deve ser totalmente reescrita, ou seja, não há possibilidade de reutilização de código. O que pode ser aproveitado é a lógica da camada, ou seja, a lógica de seus componentes. Se a aplicação *Web* construída com *software* livre não seguir a divisão do *software* em três camadas, a função do componente deve ser colocada dentro do próprio código HTML. Da mesma forma, o código não poderá ser reutilizado.

Camada de Acesso ao Banco de Dados

A camada de acesso ao banco de dados é praticamente toda escrita em linguagem SQL, existindo duas formas de realizar a migração: se a migração for para uma aplicação que utiliza como ferramenta o PHP, a camada pode ser totalmente reutilizada. Isso somente é possível se for mantida a mesma lógica da aplicação proprietária; se a aplicação utilizar como ferramenta de desenvolvimento o Zope, existem pequenas modificações nas comparações com parâmetros.

A migração da base de dados de um sistema de gerenciamento MS SQL Server 2000 e Oracle para MySQL ou PostgreSQL não apresenta maiores dificuldades, se observados alguns aspectos. O sistema de gerenciamento de banco de dados MySQL não suporta, por exemplo, relacionamentos entre tabelas, e o conceito de chave estrangeira só existe para manter a compatibilidade com outros bancos de dados, mas não é implementado. Se a aplicação construída com MS SQL Server 2000 ou Oracle for implementada utilizando essas duas características, será necessário mantê-las via programação. Dependendo do tamanho da aplicação e do modo de utilização da chave estrangeira, não vale a pena utilizar o MySQL, pois o trabalho de programação para manter as funcionalidades do sistema será muito grande. Seria mais indicado nestes casos utilizar o PostgreSQL. Porém, um dos requisitos para as aplicações *Web* é o desempenho, que é uma característica muito forte do MySQL. Dessa forma, deve-se ponderar as vantagens e desvantagens dessa migração. Se a migração de uma base de dados MS SQL Server 2000 ou Oracle for para o PostgreSQL, não haverá maiores problemas, pois a grande maioria dos recursos disponíveis no MS SQL Server 2000 e Oracle também está disponível no PostgreSQL. Aplicações *Web* escritas em MS SQL Server 2000, que é um sistema de gereciamento de banco de dados relacional, podem ser facilmente migradas para MySQL, que também é relacional, sempre considerando os aspectos de cada implementação.

O Recria Livre

Como estudo de caso para este trabalho foi migrado o módulo de cadastro de famílias do aplicativo RECRIA para *software* livre. Como o aplicativo proprietário já está em uso, optou-se por uma interface muito semelhante, a fim de facilitar sua utilização por parte das entidades da RECRIA. O aplicativo RECRIA Livre terá como licença a GNU GPL.

Analisando o estudo feito sobre as ferramentas livres para desenvolvimento de aplicações *Web*, foram feitas as seguintes opções: como ambiente e linguagem de programação, optou-se por utilizar o Zope. Esta escolha justifica-se pelo fato de que na Rede de Atenção à Criança e ao Adolescente de Caxias do Sul existem quase cinqüenta programas assistenciais em locais físicos diferentes, sendo que essas entidades possuem páginas específicas dentro do *site*. Hoje, a manutenção desse conteúdo é feita exclusivamente por uma única pessoa. Com a

utilização do Zope, seria possível que cada entidade pudesse atualizar seu conteúdo. Para facilitar ainda mais essa divisão de tarefas, será utilizado o gerenciador de conteúdo ZCoMIX [ZCoMIX 2002]. Embora o Zope esteja utilizando ele próprio o CMF [CMF 2002] e, na sua próxima versão, o Plone [Plone 2002], escolheu-se o ZCoMIX, por ser um produto que atende plenamente às necessidades dos aplicativos e também por ser desenvolvido por uma empresa local. Como banco de dados, optou-se por utilizar o MySQL porque o banco de dados de origem, o MS SQL Server 2000, é um banco de dados relacional, não existindo, portanto, necessidade de utilizar um banco de dados objeto-relacional como o PostgreSQL. Além disso, o Sistema RECRIA Livre, em sua primeira versão, terá as mesmas tabelas que a versão proprietária, o que justifica a escolha por um banco de dados igualmente relacional. Outros motivos para utilizar o MySQL, e que foram fatores determinantes para sua escolha, foram o desempenho e a característica do aplicativo RECRIA, que não necessita de recursos avançados da linguagem SQL.

Situação Atual e Perspectivas

A Prefeitura de Caxias do Sul pleiteou junto ao BNDES o financiamento do Projeto RECRIA Livre, mostrando as vantagens de migrar a aplicação existente para *software* livre, beneficiando, assim, todos municípios brasileiros interessados em informatizar suas Redes de Atenção. O BNDES, então, resolveu apoiar e acompanhar a contrução deste projeto. Está sendo formado um grupo de trabalho em três cidades brasileiras que estão mais avançadas na contrução da Rede com suas instituições: Caxias do Sul, Porto Alegre e Maceió. Esses grupos farão reuniões periódicas para aprimorar o aplicativo e montar estratégias para a sua disseminação e manutenção. Espera-se que o aplicativo cresça em tamanho e qualidade de informações, já que os vários municípios envolvidos representam diferentes realidades e experiências. Além disso, sendo um aplicativo único, tem-se uma meta a longo prazo que é formar a "Rede das Redes", ou seja, um mapa fiel da realidade da assistência social brasileira, atualizado diariamente.

Referências Bibliográficas

CMF. (2002) "Gerenciador de Conteúdo CMF", http://cmf.zope.org, Abril.

Die, Charles. (1999) "Oracle Distributed Systems", Sebastopol: O'Reilly.

Homer, Alex et al. (2000) "Professional Active Server Pages 3.0", Rio de Janeiro: Moderna.

MySQL. (2001) "*software* de Banco de Dados MySQL", http://www.mysql.com, Setembro.

Otey, Michael; Conte, Paul. (2001) "SQL Server 2000 Developer's Guide", Berkeley: Osborne/MacGraw-Hill.

PHP. (2001) "Ferramenta Hypertext Preprocessor", http://www.php.net, Outubro.

PLONE. (2002) "Gerenciador de Conteúdo PLONE", http://plone.org, Abril.

PostgreSQL. (2002) "Tutorial do *software* de Banco de Dados PostgreSQL v7.1", http://postgresql.org/idocs, Fevereiro.

Ramalho, José Antônio. (2001) "SQL Server 2000", São Paulo: Berkeley Brasil.

Raymond, Eric Steven. (2002) "The Cathedral and the Bazaar", http://www.tuxedo.org/~esr/writings/cathedral-bazaar/, Abril.

Recria. (2001) "Rede de Atenção à Criança e ao Adolescente de Caxias do Sul", http://www.recria.org.br, Agosto.

Reis, Christian (2001). "Caracterização de um Modelo de Processo para Projetos de *Software* Livre", http://www.militux.poli.org/art/christian, Março.

ZCoMIX (2002) "Gerenciador de Conteúdo ZCoMIX", http://www.x3ng.com.br, Abril.

ZOPE. (2001) "Ferramenta Z Open Environment", http://www.zope.org, Setembro.

SAAP: Desenvolvendo e Expandindo Sistemas Usando Software Livre

Paulo F. Souza Jr. [1], Wu Man Qi[1,2], Wilton S. Alencar[1,3]

[1]Interlegis – Programa Interlegis – Senado Federal
Via N-2 Anexo E Senado Federal – 70.165-900 – Brasília –Distrito Federal

[2]UnB – Universidade de Brasília

[3]UNIP – Universidade Paulista (SUPERO)

{pfsj,manqi,wiltonsa}@interlegis.gov.br

Abstract. *This article presents the System of Support to Parliamentarian's Activity - SAAP. This system was developed by Programa Interlegis's main team, with the use of a platform entirely composed by free software. The SAAP's objective is to provide one set of integrated subsystems that allow optimization of tasks related to parliamentarian's activity. This includes mainly the management of parliamentarian's interactions with people, groups and organizations, in several Legislative Houses of the country. The SAAP was fully developed in Zope, by exploring the expansion possibility of server's features for posterior system implementation.*

Resumo. *Este artigo apresenta o Sistema de Apoio à Atividade Parlamentar - SAAP. Este sistema foi desenvolvido pela equipe central do Programa Interlegis, com a utilização de uma plataforma composta totalmente por software livre. O objetivo do SAAP é prover um conjunto de subsistemas integrados que permita a otimização das tarefas relativas à atividade parlamentar. Isso envolve principalmente o gerenciamento das interações de parlamentares com pessoas, grupos e organizações, nas diversas Casas Legislativas do país. O SAAP foi todo construído em Zope, explorando a possibilidade de expansão das funcionalidades do servidor para posterior construção do sistema.*

1. Introdução

O Sistema de Apoio à Atividade Parlamentar - SAAP - é a terceira experiência da equipe central do Programa Interlegis no uso de software livre, para se atingirem as metas do programa, no que se refere a produtos e serviços em tecnologia da informação. A primeira experiência foi o SAPL - Sistema de Apoio ao Processo Legislativo [1] e a segunda, o portal Interlegis (www.interlegis.gov.br).

O Interlegis é um programa concebido, em 1997, pelo Centro de Informática e Processamento de Dados do Senado Federal – Prodasen (atual Secretaria Especial de Informática) – e vem sendo implementado pelo Senado Federal em parceria com o Banco Interamericano de Desenvolvimento – BID. O seu objetivo é a modernização e a integração do Poder Legislativo nos seus níveis federal, estadual e municipal, promovendo maior transparência e interação desse Poder com a sociedade. Os meios utilizados são as novas tecnologias de informação (Internet, videoconferência e transmissão de dados), que permitem a comunicação e a troca de experiências

entre as Casas Legislativas e os legisladores e entre o Poder Legislativo e o público, visando aumentar a participação da população no processo legislativo.

O SAAP é composto de um conjunto de subsistemas integrados, baseados em tecnologia Web. O seu objetivo é permitir a otimização das tarefas relativas à atividade parlamentar. Isso envolve principalmente o gerenciamento das interações de parlamentares com pessoas, grupos e organizações, nas diversas Casas Legislativas do país.

A primeira versão estável do sistema está disponível no portal colaborativo do Interlegis (http://genesis.interlegis.gov.br), desde o final do ano passado, para a comunidade legislativa. Por meio desse portal, esta comunidade poderá fazer sugestões e recomendações que, uma vez analisadas e aceitas, serão incorporadas ao sistema nas suas sucessivas versões.

2. Subsistemas do SAAP

Gerenciamento de Áreas de Trabalho: esse subsistema permite a um Administrador do sistema criar uma nova área de trabalho, associando-a a um proprietário, por exemplo, um parlamentar. Uma vez criada a área, o acesso a ela é restrito ao seu proprietário e as pessoas autorizadas por ele. Na criação da área, é gerada nela a estrutura dos subsistemas de contatos, atendimentos, compromissos, documentos e mala direta.

Gerenciamento de Contatos: esse subsistema permite a um usuário autenticado, numa determinada área de trabalho, cadastrar as informações básicas de pessoas físicas e jurídicas, bem como seus endereços, telefones, endereços eletrônicos, filiações partidárias e grupos aos quais elas pertencem.

Gerenciamento de Atendimentos: esse subsistema permite a um usuário autenticado, numa determinada área de trabalho, cadastrar os processos de atendimento (solicitação) que chegam num gabinete parlamentar, facilitando o acompanhamento das suas tramitações.

Gerenciamento de Compromissos: esse subsistema permite a um usuário autenticado, numa determinada área de trabalho, agendar e acompanhar os compromissos e as tarefas do dia-a-dia do parlamentar.

Gerenciamento de Documentos: esse subsistema permite a um usuário autenticado, numa determinada área de trabalho, armazenar os documentos em diversos formatos (doc, pdf, txt, html, entre outros) numa biblioteca, onde seus conteúdos serão catalogados, possibilitando buscas por palavras do texto.

Mala Direta: esse subsistema permite a um usuário autenticado, numa determinada área de trabalho, emitir cartas, ofícios ou etiquetas elaboradas para um conjunto de pessoas selecionadas.

3. Plataforma Adotada – Software Livre

A plataforma adotada é composta totalmente por software livre:

- Sistema Operacional: Linux (distribuição Conectiva 8.0 [2])
- Servidor de Aplicações Web: Zope [3]
- Linguagem de Programação: Python [4]
- Banco de Dados Orientado a Objetos: ZODB [5]

4. Por Que Zope?

O Zope - Z Object Publishing Environment – é uma plataforma livre para desenvolvimento de aplicativos Web, que vem integrado com um servidor de banco de dados orientado a objetos ZODB,

visa criar sites e aplicações dinâmicas na Web. A sua administração é feita via navegador da Internet como Netscape ou Internet Explorer.

Essa ferramenta foi desenvolvida pela empresa Digital Creations (atual Zope Corporation), utilizando uma linguagem de alto nível orientada a objetos chamada Python. A sua distribuição está disponível para vários sistemas operacionais tais como GNU/Linux, FreeBSD, Windows 95, 98, 2000 e NT.

Uma das grandes características do Zope é o fato de ser um software com o código fonte aberto. Além disso, ele oferece diferentes componentes e funcionalidades que facilitam o desenvolvimento de aplicativos Web, tais como:

- Gerenciamento e desenvolvimento de aplicativos via Web;
- Controle de acesso ao nível de perfil de usuário;
- Catalogação de documentos e pesquisa por palavra-chave;
- Integração com banco de dados relacional;
- Suporte à comunicação com outros aplicativos via protocolo XML-RPC, SOAP, CORBA e outros;
- Possibilidade de adição de módulos desenvolvidos em Python;
- Publicação "automática" de todos os objetos criados no Zope por meio dos protocolos que ele possui suporte (HTTP, FTP, XML-RPC, WebDAV, CORBA, SOAP e outros);
- Extensão do Zope via a criação de produtos OO, semelhantes aos *packages* do Java, utilizando ZClasses ou FSP (*File System Product*).

5. Implementação do SAAP

O SAAP foi desenvolvido com a utilização de ZClasses. Essa característica permite a expansão das funcionalidades do servidor de aplicações Zope a partir da construção de componentes personalizados.

Esses componentes, uma vez construídos, estarão disponíveis para serem utilizados no desenvolvimento de qualquer sistema, podendo ser colocados à disposição da comunidade de software livre, a fim de que os mesmos sejam aperfeiçoados com a colaboração dos demais desenvolvedores da área.

A metodologia adotada no desenvolvimento do SAAP foi a implementação inicial de diversos componentes pré-especificados, que posteriormente serviram de base para a construção do sistema. Um ponto importante das ZClasses é que, a qualquer alteração numa determinada classe, todos os seus objetos instanciados serão atualizados automaticamente.

Foi incorporado ao SAAP, com pequenas adaptações, um produto chamado *Document Library*, que está disponível no www.zope.org. Esse produto armazena documentos e os cataloga para pesquisas por palavra-chave.

Um fator crítico desse sistema é em relação à segurança dos dados armazenados, já que esses representam informações particulares que dizem respeito a cada parlamentar. Para atender a esse requisito, o uso do Zope foi essencial, pois cada área de trabalho do sistema possui o esquema de controle de acesso semelhante ao do próprio Zope, além dos dados serem armazenados no seu banco de dados orientados a objetos ZODB .

6. Conclusão

O uso de ZClasses do Zope e a experiência com a adaptação de um produto nos mostraram que a possibilidade de expansão dos sistemas baseados em software livre é maior comparada àqueles baseados em plataforma fechada.

O SAAP é um exemplo dessa possibilidade, pois com o uso do Zope como servidor de aplicações e do Linux como sistema operacional base, ele se moldou, em uma arquitetura flexível e poderosa, via a expansão do Zope a partir de uma funcionalidade intrínseca, mas que é inerente ao software livre.

Essa já é a terceira experiência do uso de software livre pelo programa Interlegis no âmbito do Poder Legislativo, consolidando mais uma vez o uso dessa opção como uma ferramenta robusta e eficaz, capaz de fazer frente a qualquer ferramenta comercial convencional.

Atualmente, o Interlegis tem sido referência, no âmbito governamental, sobre o uso de software livre no desenvolvimento de soluções coorporativas, sendo fomentador desse tipo de iniciativas em outras organizações governamentais como: ABIN, Ministério do Planejamento, DataPrev e INSS.

7. Referências

[1] Wu Man Qi, “Uso de Software Livre no Legislativo: SAPL – Sistema de Apoio ao Processo Parlamentar”, Anais do III Workshop sobre Software Livre. Porto Alegre: SBC, p. 66-69, 2002.

[2] Conectiva, http://www.conectiva.com.br.

[3] Pelletier, M. and Latteier, A., “The Zope Book”, Ed. 2.5, 2002, http://www.zope.org/Members/michel/ZB/IntroducingZope.dtml.

[4] Harms, D. and McDonald, K., “The Quick Python Book”, 2000.

[5] Bwarsaw and Jeremy, “Zope Object Database”, 2003, http://www.zope.org/Wikis/ZODB.

Implantação do Boot Remoto no Laboratório de Informática da Universidade de Caxias do Sul

Fabrício Scariott[1], Luciana Soardi[1], Liliane F. Giordano[1], Alexsandre Michels[1]

[1] Laboratório de Informática – Universidade de Caxias do Sul (UCS)
Caixa Postal 1.352 – 91.001-970. Caxias do Sul – RS – Brazil

{fscariot, lsoardi, lfgiorda, amichels}@ucs.br

Resumo. *Este artigo descreve uma alternativa encontrada para resolver o problema enfrentado no Laboratório de Informática da UCS. O problema era a quantidade de tempo necessária para reconfigurar os computadores, ficando estes sem uso. A alternativa encontrada foi a utilização do Boot Remoto, no qual todos os programas, inclusive o sistema operacional, acessados pelas estações de trabalho encontram-se armazenados em um ou mais servidores. Sendo assim, o processo de manutenção dos computadores torna-se mais prático e rápido.*

Abstratct. *This article describes a solution to a problem faced by UCS' Computing Laboratory. The problem was the amount of time spent on reconfiguring the PCs when it was necessary, during this time the computer was unaccessible. The alternative found was adopting Remote Boot, in which all the programs (including the operational system) accessible by the workstations are stored in one or more servers. Thus, the computer's maintenance becomes faster and more practical.*

1. Introdução

O Laboratório de Informática da Universidade de Caxias do Sul estava encontrando dificuldades na manutenção dos computadores das salas de laboratório, devido aos constantes problemas de *software*, configuração do sistema operacional e, conseqüentemente, insatisfação de alunos e professores.

Foram estudados diversos sistemas para agilizar o processo de reinstalação de computadores nas salas de laboratório. Entre eles, um sistema utilizado pela Universidade de Passo Fundo e que também foi comentado em uma palestra de Software Livre.

Esse sistema, conhecido como Boot Remoto, consiste na implementação do boot via placa de rede. A idéia do projeto é permitir uma troca de sistema operacional apenas reinicializando o computador optando pelo tipo de sistema desejado. Desta mesma forma é possível restaurar as configurações originais do sistema operacional ou dos programas.

A seguir, são explicados os motivos que levaram à escolha do Boot Remoto como solução para a manutenção dos computadores; é feito um comparativo entre Soluções Proprietárias e Soluções Livres do problema; é apresentada uma explicação sobre o funcionamento do Boot Remoto e experiências que foram constatadas com a sua utilização.

2. Comparação entre Soluções Proprietárias e Soluções Livres

Até ser adotado o Boot Remoto, a solução para restauração de imagens[I] utilizada no Laboratório de Informática era o *software* proprietário Ghost na rede Novell, *software* de

[I] Arquivo que contém uma cópia fiel de uma instalação de Sistema Operacional completo com seus respectivos aplicativos.

clonagem de HD da Symantec. Ele tornava o processo demorado, pois necessitava de acesso à rede por um dispositivo de boot e de uma intervenção direta do técnico.

A necessidade de buscar uma nova alternativa em *software* livre para o processo de manipulação de imagens em estações de trabalho, foi o fator decisivo para o estudo do Boot Remoto como forma de substituir o Ghost.

A implantação do sistema de Boot Remoto nos computadores do Laboratório de Informática iniciou no primeiro semestre de 2002, como forma de diminuir a quantidade de computadores sem uso, em decorrência de problemas de *software* e configuração.

Inicialmente, foram encontradas algumas dificuldades na implantação do Boot Remoto: (i) Inexperiência no ambiente Linux; (ii) Adaptação do sistema existente para a realidade do Laboratório de Informática da Universidade de Caxias do Sul; (iii) Dificuldades em configurações do servidor de DHCP; (iv) Grande tamanho das imagens.

Superadas estas dificuldades, pode-se observar a praticidade e rapidez que o sistema de Boot Remoto oferecia na manutenção das estações de trabalho. Além disso, ele possibilita a escolha do sistema operacional a ser utilizado.

A implantação do Boot Remoto substituiu perfeitamente a solução adotada anteriormente como método de restauração de imagens. A Tabela 1 apresenta as vantagens e desvantagens do Ghost e dos *softwares* de manipulação de imagens.

Tabela 1. Vantagens e Desvantagens das Soluções Proprietárias e Livres.

	Ghost (Proprietário)	**Boot Remoto** (*Freeware*)	**G4U** (*Software* Livre)	**PartImage HD** (*Software* Livre)
Vantagens	• Usa protocolo IPX/SPX. • Pode-se modificar arquivos diretamente dentro das imagens.	• Permite a rápida restauração dos sistema operacional. • Facilita a manutenção das estações de trabalho.	• É uma tecnologia livre. • Disco de boot Linux. • Usa o DHCP. • É bastante seguro e prático.	• É uma tecnologia livre. • Disco de boot Linux. • É bastante rápido e seguro.
Desvantagens	• É proprietário. • A versão utilizada não permite a restauração do Linux. • Gera um grande tráfego na rede	• É *freeware*. (Bpbatch) • Usa protocolo TFTP[II]. • Gera um grande tráfego na rede.	• Precisa do disco de boot. • Usa o protocolo FTP. • É lento. • Gera um grande tráfego na rede.	• Precisa de disco de boot e o disco de root. • É pouco prático. • Gera um grande tráfego na rede.

Dentre as soluções apresentadas na Tabela 1, o *software* utilizado para a manipulação de imagens foi o Boot Remoto através do Bpbatch, devido à sua praticidade. Porém, como o objetivo do Laboratório de Informática é utilizar uma solução livre, o estudo da viabilidade de utilização de G4U e PartImage está sendo aprimorado.

[II] *Trivial File Transfer Protocol* – é um protocolo de transmissão de arquivos FTP.

3. Funcionamento do Boot Remoto

O Boot Remoto estabelece uma conexão entre uma estação de trabalho e o servidor, possibilitando a transferência de imagens de um sistema operacional qualquer para a estação. No *software Bpbatch* essa transferência é realizada através de *scripts* configuráveis, e quando há uma atualização da imagem, o servidor automaticamente a repassa para a estação. Caso haja alguma alteração no sistema da estação de trabalho, algum aplicativo desinstalado, alteração de alguma configuração ou problemas no sistema operacional, simplesmente reinicializa-se a estação e reestrutura-se o sistema operacional desejado.

Para a utilização do Boot Remoto foi necessário criar diferentes tipos de imagens, uma para cada tipo de *hardware*, para que ao baixar a imagem na estação de trabalho não ocorram problemas de detecção de placas, obrigando a configurá-las manualmente. É importante ressaltar que para cada configuração de *hardware* deverá existir uma imagem [STU00].

4. Considerações Finais

O uso do Boot Remoto nas salas de laboratório tem apresentado bons resultados, sendo que o próprio aluno pode solucionar algum eventual problema no computador que está utilizando, bastando reinicializá-lo e escolher a opção de restauração do sistema operacional. Sendo assim, o processo de recuperação e ativação da estação de trabalho é agilizado, uma vez que apenas os problemas mais graves são realmente comunicados aos técnicos, reduzindo substancialmente o atraso nas aulas causado por problemas nas estações.

Foi preciso adaptar o sistema para uma versão mais recente de Linux e também ao tipo de rede utilizada. Inicialmente, foi utilizado o Linux Red Hat 7.2 por oferecer maior praticidade na manipulação das configurações do computador e pela inexperiência no ambiente Linux. Depois do processo ter sido efetuado com êxito, o sistema operacional do servidor foi trocado para Debian 3.0, e continua até hoje, por apresentar maior estabilidade e segurança para os servidores [DEB03].

O Laboratório de Informática tem grande interesse em expor a solução do Boot Remoto para que haja uma preocupação em cima da sua utilização, principalmente, nas universidades e demais instituições de ensino, uma vez que esse tem apresentado grande praticidade e resultados positivos para o laboratório. Porém, apesar de ter-se procurado soluções no âmbito de *software* livre, ainda não foi encontrada nenhuma solução que possa ser equiparada ao *Bpbatch*.

Atualmente, o Boot Remoto está sendo utilizado em seis salas de laboratório, totalizando 135 computadores e quatro servidores de Boot Remoto. Esta tem sido uma ótima solução para o problema de manutenção dos computadores.

A Figura 1 esquematiza a organização do Boot Remoto no Laboratório de Informática na Universidade de Caxias do Sul.

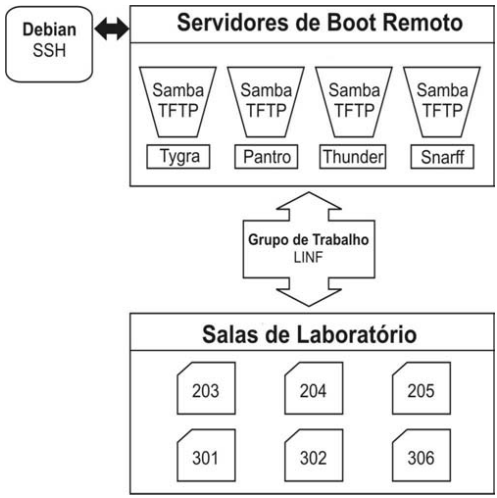


Figura 1. Organização do Boot Remoto no Laboratório de Informática da UCS.

Futuramente, pretende-se implantar o Boot Remoto em todas as salas de laboratório da Universidade de Caxias do Sul, visando assim melhorar o funcionamento dos computadores e a qualidade no atendimento aos alunos e professores.

5. Referências

[BOO03] Boot Manage. (2003) “Hot Technology”. http://www.bootix.com.

[DEB03] Debian. (2003) “Debian”. http://www.debian.org.

[FEY02] Feyrer, Hubert. (2002) “g4u - Harddisk Image Cloning for PCs”. http://www.feyrer.de/g4u/.

[GOL00] Goldschmidt Neto, Frederico Henrique. (2000) “HOWTO para criação de imagens para boot remoto”. http://upf.tche.br/~fred/remote-boot.html, January.

[ICS03] ISC Privacy Statement. (2003) “ISC Dynamic Host Configuration Protocol (DHCP)”. http://www.isc.org/products/DHCP/.

[PAR03] Partition Image project for Linux. http://www.partimage.org. (2003)

[REM00] Rembo Technology. (2000) “BpBatch home page”. http://www.bpbatch.com.

[STU00] Stückelberg, Marc Vuilleumier and Clerc, David. (2000) “Linux Remote-Boot mini-HOWTO: Configuring Remote-Boot Workstations with Linux, DOS, Windows 95/98 and Windows NT”. http://cuiwww.unige.ch/info/pc/remote-boot/howto.html, February.

Open Source Software on Web-Based Education

Arnaldo J. Bastanzo, Alejandro C. Rodriguez Costello,
Diego A. Bottallo, Federico Wiecko, Fernando A. Villar

Grupo de Desarrollo de Educación a Distancia (GDED)
Instituto Politécnico Superior "General San Martín"
Universidad Nacional de Rosario (UNR)
Av. Pellegrini 250 – Rosario – Santa Fe – Argentina

abastanz@unr.edu.ar

Abstract. *This paper describes the experience in the ILIAS open source GPL license e-learning framework, developed by the University of Cologne, Germany. This work consists of two fundamental levels. First, we have evaluated it as on-line course teaching tool (Extranet Solution) and as a complement to the attendance courses of the University Systems Analysts career (Analista Universitario en Sistemas) - (Intranet Solution). Then, we have made a verification of portability of the environment under Intel and Sun Sparc architectures, using Linux and Solaris OS, and the analysis of the development of an automatic installation tool.*

1. Introduction

The impressive development of communications and information technologies has brought up an increasing globalization of higher education. This situation is pushing forward educational institutions to collaborate and participate in complex networks for the interchange of information and experiences, by means of Web course materials, multimedia interactive software, video conferences, virtual lectures, etc.

According to the European Community Commission briefing "Web-Based Education" [1], the e-learning initiative will allow a strong cooperation between public and private sectors, between education, formation and cultural agents on one side, and content industry agents on the other.

The student as an active entity of his own learning process plays a more relevant role on e-learning than on traditional learning.

The pedagogical principle of individualization proposed by the New School, at the end of the XIX century, can be boasted on the XXI century virtual educational environments.

It must also be considered that the extensive use of information technology has generated new pedagogical paradigms [2], and a unified criterion has not been found yet.

2. ILIAS: A Web-Based Learning Management System

ILIAS is a web-based learning management system (LMS) [3], which was originally developed in the VIRTUS project at the University of Cologne, Germany [4] and now, it has become an Open Source software under GPL.

ILIAS consists of tools for learning, authoring, information access and co-operative work, thus presenting an integrated environment for learning and teaching on the Internet. ILIAS authors can create entire courses within a team and publish them on the web. Students can conform groups to work through learning material and communicate with each other or to their tutors.

The main modules of ILIAS take into account:

- Personal desktop for each user
- Learning environment with personal annotations, tests, glossary, print function, search engine
- Communication features like news system and discussion forums
- Group system to organize group members and resources
- Authoring environment (Editor) to create courses
- Context-sensitive help system for learners and authors
- System administration interface

Besides this workplace concept, some principles determined its development process. ILIAS did not follow a unique pedagogical approach; it tried to open up the system for different approaches that learning material authors can follow and, to improve the flexibility of the system in order to make it usable for authors coming from different pedagogical schools.

The production of learning material is a cost intensive process. Usually, the content consists of several multimedia components, the portability and scalability of which often is extremely costly. In order to minimize costs ILIAS supports the idea of re-using and exchanging learning material. Therefore, it implements some features that prepare the system for future compatibility to applications developed according to international standards such as IMS, SCORM, AICC and ARIADNE [5, 6].

Within this workplace, the users play different roles such as system administrator, authors, tutors (teachers) and students (learners). The system administrator puts the actors together based upon the concept of a virtual unit course by creating groups and assigning the access rights, privileges and resources needed: mailboxes, forums, discussion groups, chat rooms, documentation repositories, helpers, etc. (See Figure 1).

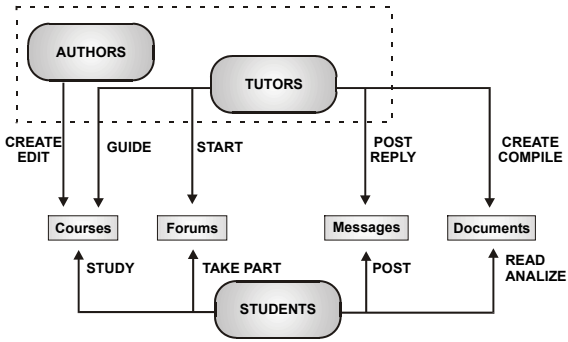


Figure 1. ILIAS environment

2.1. Technical information about ILIAS

The installation of the LMS requires a set of applications not included in the standard distribution of ILIAS, which establishes its operative framework. These applications are available under GPL license and can be obtained in source or binary formats for specifics platforms in many web sites. With the source code, the portability of the LMS to different operating systems becomes possible.

The packages needed for the installation of ILIAS version 2.3.6 (stable) are:

- Web server Apache 1.3.27 - www.apache.org
- Relational database MySQL 4.0.12 - www.mysql.com
- Server-side programming language PHP 4.3.1 - www.php.net
- Graphic library GD 1.8.4 - www.boutell.com/gd/
- Compression library Zlib 1.1.4 - www.info-zip.org/pub/infozip/zlib/
- Imaging compression library IJG JPEG 6b ftp://ftp.uu.net/graphics/jpeg/jpegsrc.v6b.tar.gz
- Graphic tool ImageMagick 4.2.9 - www.imagemagick.org
- Compression utility Info-ZIP Zip 2.3 - www.info-zip.org
- Decompression utility Info-ZIP Unzip5.50 - www.info-zip.org
- PNG library libpng 1.2.5 - www.libpng.org

Since ILIAS was developed for Linux and Sun Solaris operating systems, we have focused our work on these operating systems in particular.

To install these applications we can enumerate some highlights:

a) The Sparc Solaris operating system does not provide these tools by default. However, a lot of them are available in binary form from the Web [www.sunfreeware.com].

b) In Linux operating systems depending on the distribution in particular, these applications may be present or not. The recent main Linux distributions such as RedHat, Debian, Suse and Mandrake meet these requirements.

Once all the applications above-mentioned have been obtained, the installation process can be started.

3. Our experience in e-learning

Our first experience in e-learning began four years ago. At the beginning, we adopted the LMS WebCT environment [7]. At that time, the product was freeware, but in recent versions, it became a commercial product. At that moment, we focused our attention in investigating the philosophy of the open source project and we concluded that was possible to replace the original environment by another GNU product. We were seeking for free software products at international levels, and we selected ILIAS as our standard e-learning environment [8, 12].

For two years at the *Instituto Politécnico Superior,* we posted the University Systems Analysts entrance course using this platform. The course was dictated in both modalities: attendant and on-line.

ILIAS gave us the possibility to follow the student's acquisition of new knowledge, to obtain early diagnostics by making different kind of tests, to establish fluid communication channels, in order to answer questions and therefore guide the students and, as a side effect, to foster the student's interest in the use of free software.

Statistical evaluations have probed that in both modalities (online and attendant) the ratio of approved students is near to one. Moreover, actually we use ILIAS as a complementary documentation and communication tool for the conventional student's attendant activity. That is to say, an Intranet Solution.

At the same time, the institute offer extra-curricular courses about information technology using the e-learning modality.

4. Analysis of the development of an assisted installation

The installation tasks of the applications described above in the section 2.1 implies complexity for the majority of organizations interested in the e-learning formation, driving them in many cases to adopt "Turn Key" commercial systems.

To minimize the installation problems, and to increase the participation of users, we proposed the following items: (1) to define an operative platform, (2) to provide the means needed to build a certifiable operative environment.

We denominate this certification ISRP (ILIAS System Ready Platform). Only when hardware equipment with its operating system complies the ISRP certification statements, the installation of LMS ILIAS can be started.

For this reason we focussed in the generation of packages, such as RedHat Package Manager [9], Debian Package [10], Sun Cobalt Package (BlueLinQ) [11]. And this in turn has gave us the possibility to aim a great variety of operating platforms such as Solaris (for Sparc and Intel) in versions 7, 8 and 9, Sun Cobalt Linux, and the Linux distributions Redhat, Debian, Suse and Mandrake.

5. Discussion and Conclusion

In Education, Open Source or Proprietary Systems? An answer to this is that the university teachers and researchers' responsibility on educational affairs, considering their influence on society, is to avoid that communication and information technology world may become a business and technological monopoly.

HackAdemia: Uma Experiência com Preparação de Novos Desenvolvedores para a Comunidade Software Livre

Francisco J. Monaco[1], **Celso R. G. Providelo**[2], **Patricia Bilotta**[2]

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo
Av. Trabalhador São-Carlense, 400 - Centro 13560-970 São Carlos, SP

[2]Departamento de Engenharia Elétrica - EESC — Universidade de São Paulo
Av. Trabalhador Sãocarlense, 400 - Centro 13566-590 São Carlos, SP

[3]Departamento de Hidráulica e Saneamento - EESC — Universidade de São Paulo
Av. Trabalhador Sãocarlense, 400 - Centro 13560-250 São Carlos, SP

monaco@icmc.usp.br, cprov@sel.eesc.usp.br, pbilotta@sc.usp.br

Abstract. *Among participants of the international community of Free Software developers it is a curious fact, even intriguing, the noticeable enthusiasm with which so many volunteer contributors spontaneously get involved in projects which assign them high demanding responsibilities in terms of time and mental efforts — in fact, with such a commitment that not rarely conciliating this activity with other regular duties become a challenge of difficult solution. The experiment entitled HackAdemia (*HackAdemy*), conceived under such motivation, has shown to be an useful instrument for the exploration of this potential in the context of computer sciences teaching. Working as a source of experiences for researches on both conceptual and methodological issues associated to free software development model, the project contributes to the training of new talents disposed to qualify professionally for the new work field opened by its dissemination.*

Resumo. *Entre os participantes da comunidade internacional de desenvolvedores de software livre é fato curioso, intrigante até, o notório entusiasmo com que tantos colaboradores voluntários espontaneamente envolvem-se em projetos que lhes atribui compromissos de considerável demanda em termos de tempo e esforço intelectual — de fato, com tal empenho que não raro, conciliar essa atividade com demais obrigações regulares torna-se desafio de difícil solução. O experimento apelidado HackAdemia, concebido sob tal motivação, tem se revelado um instrumento útil na exploração desse potencial no contexto do ensino de computação. Funcionado como fonte de experiências para a pesquisa em aspectos conceituais e metodológicos associados ao modelo de desenvolvimento de software livre, o projeto contribui para a capacitação de novos talentos dispostos a qualificarem-se para atuar profissionalmente no campo de trabalho que se abre mediante sua disseminação.*

1. Introdução

Após duas décadas ofuscado pela preponderância do mercado de software proprietário, o Movimento Software Livre (MSL) re-emerge hoje como um desafiante paradigma dentro dos moldes tradicionais vigentes na indústria, propondo um modelo de desenvolvimento baseado no compartilhamento irrestrito do conhecimento e acesso não seletivo à tecnologia. O notado sucesso de empreendimentos realizados sob seus princípios de trabalho, a exemplo dos projetos GNU e Linux, chegando ao temário das conferências acadêmicas, suscita, então, o interesse na investigação

de metodologias adequadas às suas intrigantes peculiaridades, onde destacam-se abordagens de auto-coordenação descentralizada entre equipes autônomas heterogêneas, processos evolutivos concorrentes, modelos de negócios que preterem o direito privilegiado à propriedade intelectual em favor do potencial colaborativo e formas de organização de trabalho que substituem a disputa competitiva pelas práticas de produção cooperativa [Rodrigues et al., 2001, Monaco et al., 2002].

O experimento apelidado *HackAdemia* ("Hacker Academia") , realizado sob os auspícios do Projeto GNUsp[1] na Universidade de São Paulo, e cuja primeira fase foi implantada durante o ano de 2002, é dedicado à exploração dessas questões dentro do contexto acadêmico, contribuindo para a capacitação de novos talentos dispostos a qualificarem-se para atuar profissionalmente no campo de trabalho que se abre mediante a disseminação do Software Livre.

2. Objetivos

Entre os participantes da comunidade internacional de desenvolvedores é fato curioso, intrigante até, o notório entusiasmo com que tantos colaboradores voluntários espontaneamente envolvem-se em projetos que lhes atribui compromissos de considerável demanda em termos de tempo e estudo — de fato, com tal empenho que não raro, conciliar essa atividade com demais obrigações regulares torna-se desafio de difícil solução. Para muitos estudantes, em especial para aqueles motivados pela área de computação, em cujo meio a iniciativa desperta-se precocemente em função da própria natureza do objeto de trabalho, a interação com pares de interesses afins, a gratificante experiência de reconhecer-se como responsável por um esforço bem sucedido e a realização de ser notado pelo seu mérito individual têm proporcionado não apenas estímulo para aprimorar conhecimentos, mas também valiosa oportunidade para ganhar auto-confiança e exercitar talentos — quer habilidades técnicas, quer caracteres de liderança e disposição para trabalho em equipe.

O projeto HackAdemia, concebido sob tal motivação, tem se revelado um instrumento útil na exploração desse potencial no contexto do ensino de computação. Originalmente dirigido a um conjunto restrito de participantes diretamente ligados à organização da atividade, o experimento, na sua segunda fase, a iniciar-se no segundo semestre de 2003, objetiva estender sua abrangência à comunidade extra-universitária e acolher qualquer interessado em iniciar-se na Comunidade Software Livre (CSL) internacional como colaborador. O lançamento planejado para este ano se dará na forma de uma série de chamadas de trabalhos, a serem divulgadas de forma programada em escopos específicos, como na edição anterior. Em seu anúncio, os interessados são convidados a indicarem um software original, uma nova idéia ou um projeto antigo que estão dispostos a por em prática. Para tal, devem apresentá-lo na forma de uma proposta, materializando objetivos, vantagens e condições de viabilização, e defendê-la perante a coordenação e demais participantes do projeto, a fim de encorajar a formação de uma equipe de contribuidores voluntários. Como contrapartida, HackAdemia se propõe a encubar o projeto em sua infra-estrutura de recursos, e oferecer o apoio ao seu alcance para dar suporte à execução do trabalho. Atualmente esse benefício pode incluir a hospedagem dos projetos nos servidores do GNUsp, facilidades on-line como disponibilização de repositório CVS, ferramentas de depuração, site Web, lista de discussão etc., bem como o oferecimento de seminários e mini-cursos sob tópicos específicos, conforme as necessidades técnicas dos participantes.

Além de investir na capacitação de novos desenvolvedores e revelar novos "hackers[2]", e

[1] O Projeto GNUsp (`www.gnusp.org`) é uma iniciativa científico-educacional liderada por professores, estudantes e contribuidores voluntários cujo objetivo é promover o conhecimento acerca de fundamentos conceituais e técnicos em Software Livre entre usuários e desenvolvedores profissionais. Desde seu início, o Projeto tem suportado diversas atividades no campus universitário, incluindo a organização de cursos e seminários e manutenção de recursos para suporte à comunidade de usuários.

[2] Onde o termo refere-se ao especialista de desenvolvidas habilidades técnicas e talento, conforme significado originalmente consagrado entre praticantes da área.

de contribuir para o aprimoramento da coleção de software livre, o objetivo essencial do projeto HackAdemia é didático. Por isso, à cada projeto encubado é indicado um tutor: sua tarefa é orientar os desenvolvedores aprendizes e proporcionar-lhes suporte em seu aprendizado. Tendo como referência o Projeto GNU [GPL, 1991], os trabalhos do GNUsp — e portanto o HackAdemia — identificam-se com suas abordagens filosóficas e metodológicas, incorporando-as no programa de treinamento. Por isso, dentre as regras vigentes para todos os grupos de trabalhos incluem-se as de que nenhuma forma de tecnologia proprietária pode ser incluída em qualquer projeto e a de que todo artefato de software produzido é obrigatoriamente distribuído sob a licença GNU GPL[3] (*GNU General Public License*). Os projetos são continuamente avaliados pelos coordenadores e alguns critérios mínimos de qualidade e de atividade são cobrados para garantir a continuidade do suporte.

2.1. Resultados

Na primeira fase do programa-piloto, uma chamada de trabalhos foi divulgada de forma discreta entre os alunos do campus. Dentre os que atenderam ao chamado, um grupo de voluntários, de número ainda restrito, foi selecionado para participar do ensaio preliminar. Em virtude do pequeno número de participantes naquela edição experimental do programa (cerca de 15 alunos entre graduandos e pós-graduandos), bem como da heterogeneidade de habilidades técnicas entre eles, apenas dois projetos foram capazes de convergir os interesses e propiciar a formação de equipes de trabalho. Dentre ambos, o projeto intitulado *Coolcalc* revelou-se particularmente interessante e, por isso, recebe atenção especial neste artigo.

2.1.1. Coolcalc

Coolcalc é uma calculadora científica de notação polonesa reversa. O software fornece a funcionalidade convencionalmente encontrada em aplicações similares, incluindo operações aritméticas elementares, cálculo vetorial, funções trigonométricas etc. (embora não se tivesse a pretensão de completar sua implementação no prazo previsto de um ano). A metodologia adotada na sua execução baseia-se na reprodução, em pequena escala, das práticas de trabalho tradicionalmente encontradas na Comunidade SL. Os voluntários receberam treinamento básico em linguagem C, GNU Autotools (autoconf/automake), sistema CVS, gcc/glibc e tiveram a oportunidade de praticar seus conhecimentos em exercícios preliminares.

Lançada a fase de desenvolvimento, após um período inicial de exitação — justificável pela inexperiência da maioria dos participantes nesse tipo de atividade — seguiu-se um rápido entrosamento entre a equipe, estimulada a manter interação freqüente entre os membros e conservar o estilo informal e pessoal de conversação que caracteriza a forma tradicional de comunicação da comunidade de desenvolvedores. Em tempo razoável, a equipe foi capaz de reconhecer as diferentes habilidades de cada um dos seus integrantes e, a partir daí, auto-gerir-se de forma conveniente: os mais adiantados passaram a coordenar as atividades mais críticas, sob a aprovação daqueles menos treinados — ao passo que estes voluntariamente passaram a se reunir em times responsáveis pelas tarefas para as quais se consideravam aptos. Durante toda atividade, no entanto, todos os participantes foram incentivados a assumir uma atitude de independência em relação aos coordenadores e aos colegas, buscando ainda assim manter o espírito de equipe e cooperação. Segundo essa prática, as decisões acerca de detalhes importantes na especificação e implementação de cada uma das partes têm de ser exaustivamente discutidas em aberto até atingir-se um consenso, cabendo ao líder do respectivo módulo a decisão final.

Na prática, Coolcalc foi dividido em um conjunto de módulos compreendendo a interface, as estruturas de dados principais, funções internas e os pacotes de operações (aritméticas,

[3] A opção pela GNU LGPL não é atualmente permitida.

estatísticas etc.). Cada um destes foi subdividido em módulos menores, os quais eram requisitados aos líderes pelos contribuidores de cada unidade.

Ao final do período programado, Coolcalc encontrava-se em adiantado estágio de desenvolvimento, embora a lista de *bugs* por serem sanados mantinha-se consideravelmente longa. Os dados fornecidos pelos *logs* do repositório CVS, todavia, demonstram a efetiva participação da equipe e apresenta os típicos picos de atividade permeados por interregnos marcados por modificações esporádicas (não surpreendentemente guardando previsível coincidência com os períodos de tranqüilidade e inquietação do ano letivo acadêmico). O resultado mais importante, todavia, revelou-se na forma do aprendizado proporcionado aos alunos — dentre os quais, alguns com aproveitamento excepcional — e nas lições angariadas para a próxima edição do projeto. Diversos pontos falhos foram identificados, tais como a necessidade de um melhor balanceamento entre desenvolvedores experientes e iniciantes em uma mesma equipe: grupos formados exclusivamente por neófitos encontraram dificuldades em acompanhar o ritmo do projeto, ao passo que times compostos apenas por *experts* eram sobrecarregados. A intervenção dos coordenadores na reorganização das equipes mostrou-se oportuna, uma vez que, por afinidade, os participantes tendem a se organizar em "redutos" de novatos e veteranos. Outra dificuldade anotada concerne à escolha de metodologias de engenharia de software: enquanto modelos, técnicas e ferramentas sofisticadas são as vezes preferidos pelos desenvolvedores experientes, procedimentos complexos tendem a intimidar iniciantes e desmotivar sua iniciativa na interação colaborativa; o compromisso entre ambos aspectos revelou-se em diversas ocasiões no cerne de vigorosas discussões entre os participantes.

3. Conclusões

O projeto HackAdemia tem funcionado como fonte de experiências para a pesquisa em aspectos conceituais e metodológicos associados ao modelo de desenvolvimento de software livre. Espera-se que o aprendizado ao longo da sua continuidade proporcione subsídios para novas conclusões acerca de suas características, e continue a estimular a profícua discussão acerca dos seus fundamentos técnicos e filosóficos.

Além da constatação do seu potencial no aproveitamento do crescente interesse pelo software livre entre os alunos e da sua efetividade como catalizador de iniciativas mediante sua exploração como atividade acadêmica, o experimento tem se revelado capaz de fomentar o desenvolvimento de projetos reais — úteis não somente através da experiência didática proporcionada, mas também diretamente na forma de aplicações de funcionalidade prática e com qualidade. Basta lembrar que os exemplos mais representativos do software que está revolucionando a história da computação surgiram através das mãos de estudantes e voluntários.

Referências

GPL (1991). The gnu project and the free software foundation. Free Software Foundation. http://www.gnu.org.

Monaco, F. J., Providelo, C. R. G., Rodrigues, W. J., and Borges, J. H. G. (2002). Experiences with the concept of free software in a professional education environment. In *Workshop Software Livre 2002 (SBC), III International Forum on Free Software*, pages pp. 78–81, Porto Alegre, Brazil.

Rodrigues, W. J., Monaco, F., Gonzaga, A., and Aguiar, M. L. (2001). Free software in superior education : Talking about freedom to future professionals. In *Workshop Software Livre 2001 (SBC), II International Forum on Free Software*, pages pp. 43–45, Porto Alegre, Brazil. Sociedade Brasileira de Computação (SBC).

Kiwi: Um Framework Alternativo para Aplicações com Interface Gráfica

Christian Robottom Reis[1]

[1] Async Open Source
R. Dona Alexandrina, 2534
São Carlos, SP
13566-290
<kiko@async.com.br>

Resumo. *Kiwi é um framework de classes em Python que tem por objetivo simplificar a criação de aplicações com interfaces gráficas. Ao invés de oferecer um ambiente de desenvolvimento e geração de código, no entanto, Kiwi inclui classes de alto nível que representam bem características comuns de aplicações desta natureza, como janelas e formulários complexos, listas reordenáveis, campos de entrada validados e auto-completados, entre outras. Kiwi se baseia nas excelentes bibliotecas livres GTK+, PyGTK e libglade, e é licenciado como software livre através da GNU LGPL.*

Abstract. *Kiwi is an application framework written in Python, designed to ease the development of applications with graphical user interfaces. Instead of offering a code generation or development environment, however, Kiwi offers high-level components that match closely common aspects of this type of software, including application windows, complex forms, reorderable lists, and validated and autocompleted data entry components. Kiwi is based on excellent (and free) libraries GTK+, PyGTK and libglade, and is licensed as free software under the GNU GPL.*

1. Introdução

Bibliotecas orientadas a objeto constituem um recurso interessante para estabelecer e concretizar soluções reusáveis em projetos de software. Em particular, aplicações gráficas normalmente utilizam bibliotecas de componentes — XView, Swing, Motif e GTK+, entre outras — que abstraem os elementos gráficos básicos: janelas, botões e campos de entrada de dados.

Seguindo esta mesma tendência para abstração, *frameworks* de aplicação orientados a objeto [1] representam importantes ferramentas para simplificar a criação de aplicações completas, tendo como fundamento conceitos de projeto de software bem estabelecidos. Frameworks de aplicação normalmente oferecem um padrão arquitetural, utilizado para estruturar e modularizar uma implementação.

Grande parte das aplicações com interface gráfica atualmente utiliza um projeto arquitetural derivado do padrão *Model-View-Controller* (MVC) [2]. Este padrão se baseia em três classes que representam os aspectos essenciais de uma interface gráfica: dados, apresentação e comportamento. Inicialmente criado para o ambiente Smalltalk-80, este padrão é atualmente bastante popular, sendo utilizado em inúmeras bibliotecas e aplicações.

1.1. Aspectos e Limitações do MVC

No modelo MVC, cada interface é fatorada em três classes com responsabilidades particulares:

- ***Model* (Modelo)**, que representa o dado ou conjunto de dados sendo manipulado naquela interface.
- ***View* (Visão ou Vista)**, que descreve a aparência gráfica daquela interface.
- ***Controller* (Controlador)**, que determina o comportamento da interface gráfica com base em eventos e mensagens geradas no uso da aplicação.

Embora seja de fato popular, alguns trabalhos apresentam críticas [3, 4] ao uso do modelo MVC como padrão para estruturar aplicações. Na prática, o acoplamento resultante entre as classes ainda é alto; o controlador e a visão dependem intrinsicamente um do outro, e tende a ser intensa e passagem de mensagens entre eles. É importante ressaltar que MVC foi desenvolvido para modelar componentes individuais (controles ou *widgets*) e não aplicações inteiras, sugerindo que possam ser encontradas arquiteturas mais eficazes.

1.2. GTK+, PyGTK e Libglade

Um conjunto de excelentes bibliotecas de componentes gráficos existe como software livre. Entre estas, destaca-se o **GTK+**, implementado inicialmente para a aplicação The Gimp, e utilizado em diversos aplicativos livres (por exemplo, em todo o ambiente Gnome[1]). GTK+ é um conjunto genérico de componentes gráficos, implementado em C, mas que possui bibliotecas de adaptação para outras linguagens; para Python, a biblioteca se chama **PyGTK**.

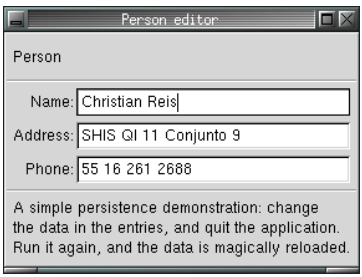

Person editor
Person
Name:
Christian Reis
Address:
SHIS QI 11 Conjunto 9
Phone:
55 16 261 2688
A simple persistence demonstration: change the data in the entries, and quit the application. Run it again, and the data is magically reloaded.

Figura 1: Exemplo de aplicação escrita usando Kiwi. O código-fonte completo é exposto na seção 2.3.

O framework Kiwi utiliza estas bibliotecas fundamentais para a implementação das interfaces gráficas propriamente ditas, mas agrega classes de alto nível para tornar o processo de construção da aplicação mais simples. Por exemplo, Kiwi oferece uma maneira bastante conveniente de utilizar a biblioteca **libglade** para definir as interfaces em tempo de execução. Através do libglade uma especificação da interface, codificada em XML, é lida e as janelas e componentes são criados dinamicamente, removendo a necessidade de se implementar em código sua definição.

2. Kiwi: Aspectos do Framework

Embora seja uma biblioteca relativamente extensa, Kiwi oferece um conjunto simples de classes para serem utilizadas em aplicações, juntamente com componentes convenientemente especializados, discutidos nas seções a seguir.

2.1. Modelo Arquitetural

Para maximizar reuso, Kiwi se utiliza do conceito fundamental do MVC: desacoplar apresentação dos dados sendo apresentados. O modelo de classes utilizado no framework Kiwi é uma derivação do MVC, utilizando como base classes View e Controller, com algumas extensões importantes. Na programação com Kiwi, o desenvolvedor apenas seleciona as classes que deseja usar, define uma interface para sua aplicação, e indica funções que implementam comportamentos especiais.

[1] Gnome é um projeto para criar um ambiente *desktop* completo; ver http://www.gnome.org/

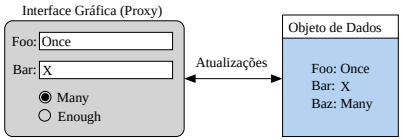
2.1.1. *Delegates*

Assim como no Swing [5] em Java, Kiwi oferece o conceito de *User Interface Delegates*. Um *delegate* reúne funcionalidade de Controller e View em uma única classe, resolvendo o problema de alto acoplamento entre as classes (através de fusão).

Um delegate normalmente representa uma janela ou outro elemento *container*. Todo delegate tem a opção de ter sua interface construída em código Python ou utilizando um arquivo XML processado pelo libglade. A alternativa libglade se mostra particularmente interessante por diversos motivos: facilidade de manutenção (através da aplicação Glade), redução do código-fonte, criação WYSIWYG (acrônimo para *What you see is what you get*).

2.1.2. *Proxies*

Proxies foram criados especialmente para o Kiwi, com base em idéias apresentadas por Allen Holub [4]. Um proxy vincula à interface de um Delegate um objeto de dados (ou do domínio), e faz com que alterações feitas na interface sejam transparentemente propagadas para o objeto do domínio, e vice-versa. O código resultante é mínimo, e por este motivo proxies são particularmente eficazes para a construção de formulários complexos.



2.1.3. *Slaves*

Tanto Proxies quanto Delegates simples podem ser criados na forma de Slaves. Um slave é uma parte de uma interface gráfica, normalmente composta de diversos elementos agregados por um determinado motivo (por exemplo, porque são associados a um mesmo objeto de domínio). Slaves permitem que uma interface seja composta de diversas partes criadas independentemente, permitindo reuso não apenas de janelas inteiras, mas de porções definidas.

2.2. Outros Componentes

Além destas classes, Kiwi oferece um conjunto de componentes com funcionalidade avançada, incluindo o CListDelegate, que permite que sejam dispostos objetos de dados em listas de maneira trivial, o AutoCombo, que é um campo de entrada de texto com auto-compleção, e o Validator, que vincula um conjunto de atributos a funções de validação.

2.3. Implementação

Kiwi é implementado na linguagem Python[2], sendo distribuído como um pacote facilmente instalável via Python Distutils. É licenciado pela licença GNU LGPL, podendo ser utilizado em qualquer projeto desde que alterações ao seu código sejam redistribuídas.

Como um exemplo simples, a seguir está apresentado o código-fonte de uma aplicação que exibe uma janela com três campos de entrada, conforme representado na figura 1. Note que o arquivo XML, que especifica a interface, não está representado, tendo sido criado com o Glade, um editor gráfico de interfaces.

[2]Uma linguagem orientada a objetos dinâmica; ver `www.python.org`.

```
from Kiwi import mainquit                  # Carregando módulos do Kiwi
from Kiwi.FrameWork import PickedModel
from Kiwi.Proxies import GladeProxy

class Person(PickledModel): pass           # Classe para o objeto de dados

person = FrameWork.unpickle_model(Person)  # Carrega dados do objeto de arquivo
widgets = [":address", ":name", ":phone"]  # Especifica os campos de entrada

view = GladeProxy(person, "Person", widgets=widgets,   # Cria instancia de Proxy,
                          delete_handler=mainquit)     # definindo campos e saída

view.show_and_loop()                       # Executa a aplicação

person.save()                              # Salva alterações em arquivo
```

3. Conclusões

Existem ainda importantes contribuições para serem realizadas na área da arquitetura de aplicações gráficas; frameworks orientados a objetos têm grande potencial para acelerar a implementação de aplicações completas, sem ocupar o desenvolvedor de tarefas enfadonhas e código repetitivo. Há também interesse de diversas organizações em adotar plataformas e bibliotecas livres no desenvolvimento de suas aplicações. A qualidade e popularidade das soluções de bibliotecas gráficas livres, que incluem o GTK+ e o Qt, sugere que é um bom campo para investigação e desenvolvimento de novas tecnologias.

Neste contexto, o framework Kiwi oferece valor não só por estar amplamente disponível, quanto por implementar um conjunto de técnicas orientadas a objeto relativamente inovadoras, que podem ser estudadas e aprimoradas gradualmente. A plataforma de aplicações comerciais Stoq[3] tem utilizado Kiwi com grande sucesso, e sua equipe está comprometida com sua evolução.

3.1. Agradecimentos

Entre outras pessoas, Andreas Kostyrka, Guilherme Salgado, Jon Nelson, Johan Dahlin e Ricardo Froelich fizeram contribuições significativas que foram integradas à base de código do Kiwi.

O desenvolvimento da biblioteca Kiwi é apoiado pela Async Open Source, empresa dedicada ao desenvolvimento de soluções em software livre. Parte deste trabalho contou com apoio Fapesp através de incentivo PIPE, vinculado ao projeto Stoq, processo nº 2001/08534-7.

Referências

[1] FAYAD, M.; SCHMIDT, D. C. Object-Oriented Application Frameworks. *Communications of the ACM*, v. 40, n. 10, 1997.

[2] G. Krasner and S. Pope. A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 system. *Journal of Object Oriented Programming*, v. 1, n. 3, p. 26–49, 1988. Disponível em: <http://www.ccmrc.ucsb.edu/ stp/publs.html>.

[3] EVERS, M. Adaptability problems of architectures for interactive software. In: *Object-Oriented Technology, ECOOP'99 Workshops, Panels, and Posters*. Lisbon, Portugal: Springer, 1999. Disponível em: <http://www.ub.utwente.nl/webdocs/ctit/1/00000017.pdf>.

[4] HOLUB, A. Building user interfaces for object-oriented systems, Part 1. *JavaWorld*, 1999. Disponível em: <http://www.javaworld.com/javaworld/jw-07-1999/jw-07-toolbox.html>.

[5] FOWLER, A. A Swing Architecture Overview. *java.sun.com*, 2003. Disponível em: <http://java.sun.com/products/jfc/tsc/articles/architecture/>.

[3] Stoq é um projeto para a criação de soluções de automação comercial em software livre; detalhes podem ser obtidos em `http://www.stoq.com.br/`

Desenvolvimento de um sistema de administração acadêmica para IES[1] utilizando-se Software Livre

Bruno R. Silva, Éverton H. Freitas, Renato A. C. Silva, André G. Santos, Ciro M. Santos, Ulisses A. Leitão

Centro Doctum de Tecnologia
Faculdades Integradas de Caratinga
Caixa Postal 35.300-037 - Caratinga - MG

{brunors, ehf, rac, andre, ciro, uli}@spep.com.br

Abstract*. The Ad*χ *system presented here is a general management tool for all the departments of a college. The purpose is to manage the relation between the student and college in a quick way, either in the academic, administrative or financiall area. Its modular structure, based on Object Oriented Programming, was designed with a Web interface that allows to execute it at several distinct environments and operation system. The tecnology used to develop it uses PHP [1] Programming language, MySQL [2] Data Base and Apache Web Server [3] on GNU/Linux[2] environment, whose licenses are GPL[3].*

Resumo. *Este artigo apresenta o desenvolvimento e as características de um sistema de administração acadêmica, batizado de Ad*χ *(Sistema X de Administração Acadêmica), visando agilizar e gerenciar todo o relacionamento do aluno com a IES, seja este na esfera acadêmica, administrativa ou financeira. Sua estrutura modular utiliza Programação Orientada a Objetos e possui uma interface Web, permitindo-lhe autonomia de execução em multiplataformas. A tecnologia usada para o desenvolvimento faz uso da linguagem de programação PHP [1], banco de dados MySQL [2] e Servidor Web Apache [3] em ambiente GNU/Linux[2], cujas licenças de uso são GPL[3].*

1 Introdução

Constituída como um dos principais centros de educação e cultura da sociedade, as IES apresentam-se como instituições sociais cuja função é o desenvolvimento de novas gerações de profissionais cidadãos capazes de contribuir para o desenvolvimento da comunidade e o crescimento pessoal. Condição para uma formação acadêmica que atenda a estes quesitos exige que os conteúdos programáticos e sua dinâmica de aprendizagem sejam estruturados com caráter de processo de formação, fornecendo a seus estudantes não somente os elementos cognitivos do processo ensino-aprendizado, mas também, formação teórico-prática e valores culturais em sentido geral.

Dois marcos na implantação do Curso de Ciência da Computação da FIC exemplificam este princípio: *i)* a experiência de uso exclusivo de software livre no ensino de graduação [4,5] iniciada há quatro anos, e *ii)* implantação do Centro Doctum de Tecnologia, como laboratório de desenvolvimento de aplicativos e sistemas para as instituições parceiras: FIC-Faculdades Integradas de Caratinga e Faculdades Doctum. Tais iniciativas tiveram o mérito de gerar e consolidar na IES um ambiente educacional voltado para a pesquisa e o desenvolvimento tecnológico.

[1] Instituição de Ensino Superior.
[2] O projeto GNU foi lançado em 1984 para desenvolver um sistema operacional completo e livre, similar ao Unix.
[3] General Public License (Licença Pública Geral).

O presente artigo descreve o desenvolvimento e as características de um sistema de administração acadêmica, batizado de Adχ (Sistema X de Administração Acadêmica), um sistema completo de gerenciamento administrativo de Instituição de Ensino Superior desenvolvido no Centro Doctum de Tecnologia. O sistema é independente de plataforma, tanto de hardware como de software, pois está sendo desenvolvido com base em uma tecnologia Web Cliente/Servidor. Seu objetivo é integrar todos os departamentos administrativos, tais como: Bibliotecas, Tesouraria, Recursos Humanos, Diretorias, Secretarias e Coordenação de Cursos.

A tecnologia usada para o desenvolvimento faz uso da linguagem PHP [1], banco de dados MySQL [2] e Servidor Web Apache [3] em ambiente GNU/Linux, cujas licenças de uso são GPL (Software Livre). As principais vantagens do projeto para a IES são: custo relativamente baixo de desenvolvimento e manutenção e flexibilidade, tendo em vista a forma customizada com que o sistema é desenvolvido.

2 Trabalhos Relacionados

O Sistema Aberto de Gestão Unificada, SAGU [6], é desenvolvido pela UNIVATES/RS utilizando ferramentas GPL e visa ser um sistema que agiliza e gerencia todo o relacionamento de um aluno com a instituição de ensino. Entretanto, o SAGU foi desenvolvido para atender às necessidades específicas da Univates. Estabelecido o contato com sua equipe de desenvolvimento, observou-se que o mesmo não possuía documentação suficiente para darmos continuidade ao desenvolvimento e não oferecia garantia de suporte.

Após uma longa análise das diferentes alternativas, a FIC, em parceria com o Centro Doctum de Tecnologia, optou pelo desenvolvimento próprio de um sistema alternativo ao SAGU, com suporte garantido.

3 Contribuição

Integrando-se os departamentos da IES em apenas um sistema, além de possibilitar uma administração eficiente, proporciona aos funcionários acesso indireto a informações de outros departamentos.

Com sua característica de acesso remoto, possibilita também à administração da IES acessar em tempo real, as informações de outros *campi* que estão separados geograficamente.

Os alunos podem acessar, através da Internet, suas informações acadêmicas, como histórico escolar, informações financeiras como débitos, consulta e reserva de livros, estrutura curricular, ementa de disciplinas, grade horária, calendário escolar e outras funcionalidades.

E por fim, a economia no que diz respeito às ferramentas de desenvolvimento, pois o servidor Web Apache, o interpretador PHP e o SGBD e o sistema operacional Linux utilizando no servidor, são de distribuição gratuita. Ou seja, todos os recursos financeiros para o desenvolvimento foram direcionados à equipe de desenvolvimento e cursos de capacitação. Além de aproveitar mão-de-obra mais acessível, pois a equipe de desenvolvimento foi formada por alunos e ex-alunos do curso de graduação em Ciência da Computação.

4 Estrutura Metodológica

Visto a grande complexidade do gerenciamento de uma IES, optou-se por desenvolver o sistema *Adχ* em módulos, que proporcionam um desenvolvimento mais simplificado e de forma a liberar para uso institucional os módulos do sistema à medida que fossem concluídos.

Com essa visão, dividiu-se o sistema nos seguintes módulos: módulo de biblioteca, módulo vestibular, módulo acadêmico, módulo pedagógico, módulo tesouraria e módulo de recursos humanos, sendo que todos estes módulos interagem entre si de forma dinâmica e contínua.

Além da divisão em módulos, o sistema reconhece vários tipos de usuários, que farão uso apenas dos módulos que possuem permissão de acesso.

O sistema fornece acesso a todos os módulo através de uma tela principal, que é acionada após o login de usuário.

O Sistema *Adχ* possui um módulo de abstração de base de dados que permite a migração para diversos SGBD's, que consiste na utilização da classe PEAR-DB[7], responsável pela abstração. Adotou-se por padrão, o uso do SGBD MySQL, mas com essa abstração é fácil migrar para Oracle, PostgreSQL e SQL Server 2000, como mostra a figura 1.

Figura 1. Diagrama de blocos

Todo o sistema é desenvolvido utilizando uma tecnologia voltada para Web, possibilitando o acesso remoto pelos seus usuários. Além disto, por não necessitar de instalação em máquinas cliente, cria uma característica multiplataforma cliente/Servidor, podendo rodar em qualquer sistema operacional, diferentes configurações de hardware, necessitando apenas de um navegador Web padrão (Internet Explorer, Nestcape, Mozilla, Mosaic etc.). Esta característica foi alcançada devido ao uso da linguagem PHP em conjunto com o servidor Web Apache.

Temos então, que o sistema *Adχ* proporciona uma ambiente multiplataforma, com economia nos recursos computacionais e acesso remoto para seus usuários.

5 Desenvolvimento

5.1 Módulo de acesso

Apesar do sistema *Adχ* ser modular, a interface do usuário com cada um dos módulos se apresenta de forma transparente, disponibilizando apenas as funcionalidades as quais o usuário tem acesso, com suas devidas permissões.

5.2 Módulo Biblioteca (Bibtec)

O módulo de Bibtec visa gerenciar a biblioteca, integrando o processo de empréstimo com os outros departamentos da IES. Basicamente existem relacionamentos com o módulo de Tesouraria e Acadêmico. Com a implantação do módulo, os alunos podem consultar o acervo através da Internet, efetuar reservas e verificar prazos de devolução. O módulo possibilita também a automação do empréstimo, com a utilização de leitores de código de barras e integração com mini-impressoras, além da total compatibilidade com o padrão Marc 21.

5.3 Módulo Vestibular

Com o objetivo de automatizar todo o processo seletivo da IES, o módulo vestibular é responsável pela inscrição online, com geração de boleto bancário, enturmação dos candidatos, geração de relatórios estatísticos, geração de etiquetas e cartões de inscrição, integração com equipamento de leitura ótica que agiliza o processo de recolhimento dos gabaritos, classificação e divulgação do resultado.

5.4 Módulo Pedagógico

O módulo pedagógico tem como finalidade possibilitar o acesso às informações pedagógicas da IES para professores, alunos e coordenadores de curso. O mesmo integra com o módulo Acadêmico, no sentido de que toda informação por ele recolhida, fica disponibilizada para uso nas secretarias de curso. Entre algumas funcionalidades, destacam-se a gestão de Cursos da IES, a gestão de disciplinas, de ementas, de grade horária e gestão de estruturas curriculares..

5.5 Módulo Acadêmico

O módulo acadêmico, pode ser considerado como o centro do sistema. Nele são gerenciados todos os aspectos referentes à gestão de notas, ausências, diário de classe online, matrícula, rematrícula e enturmação. Ele integra com o módulo Pedagógico, pois faz uso das informações por ele gerenciadas. Integra também com o módulo de biblioteca e de Tesouraria.

5.6 Módulo Financeiro

Responsável pela geração de cobranças bancárias, recolhimento de taxas extras como: multas por atraso na devolução de livros, onde verifica-se a integração com o módulo de biblioteca e emissão de relatórios e gráficos estatísticos.

5.7 Módulo Recursos Humanos

Responsável pelo gerenciamento do quadro de funcionários da IES, com todas as funções necessárias, como por exemplo: folha de pagamento e manutenção do quadro de professores e suas disponibilidades de horário.

6 Conclusão

Neste artigo, procuramos demonstrar algumas das melhorias obtidas com a implantação do sistema *Adχ* na FIC -Faculdades Integradas de Caratinga.

O sistema está sendo desenvolvido utilizando software livre, sendo o Linux, o sistema operacional utilizado para implementação e execução dos programas, embora a tecnologia utilizada permita a execução em multiplataformas.

Através do sistema *Adχ* os alunos poderão obter seu histórico, fazer consultas ao acervo bibliográfico da instituição, consultar sua situação financeira, requerer matrícula entre vários outros benefícios. Os professores por sua vez, poderão manipular seu diário virtual, através da Internet, em sua própria casa. Sendo assim, o *Adχ* promoverá a maior interação possível entre instituição, alunos e professores, estreitando os laços através da tecnologia.

7 Referências Bibliográficas

[1] WELLING, Luke; THOMSON, Laura. *PHP E MySQL: Desenvolvimento Web*. Rio de Janeiro: Campus, 2001.

[2] SUEHRING, Steve. *MYSQL, a bílblia*. Rio de Janeiro: Campus, 2002.

[3] HAWKINS, Scott. *Apache web server administration e e-commerce handlook*. New Jersey: Prentice Hall, 2001.

[4] LEITÃO, Ulisses A.; dos SANTOS, André G. e NACCARATTI, Paulo R. A., *Processo de implantação de um ambiente de Software Livre na Faculdade de Ciência da Computação de Caratinga-MG*. I Workshop de Software Livre, Porto Alegre, maio de 2000, Anais.

[5] dos SANTOS, André G.; dos SANTOS, Ciro M.; TEIXEIRA, Juliana S,; KONZEN, Alexandre e LEITÃO, Ulisses A., *Experiência de uso exclusivo de software livre na graduação*. Submetido, SBC 2003.

[6] Projeto SAGU: URL: http://sagu.codigolivre.org.br.

[7] Classe PEAR-DB: URL: http://www.pear.php.net

Modelagem e Implementação do Banco de Dados do Projeto ESCUNA usando TCM e PostgreSQL

Karina dos Santos Machado[1], Leonel Pablo Tedesco[1], Greyce Nogueira Schroeder[1], Carlos Adail Scherer Junior[1], Marcelo Pereira Nunes[2]

[1]Engenharia de Computação – Fundação Universidade Federal do Rio Grande
Caixa Postal 474 – 96201-900 – Rio Grande – RS – Brasil

[2]Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681, Prédio 30, bloco 4 – 90619-900 – Porto Alegre – RS – Brasil

{karina,leoted,greyce,ijuy}@ecomp.furg.br, mnunes@inf.pucrs.br

Resumo. *O presente artigo aborda o uso de software livre no desenvolvimento de um banco de dados para o controle de uma rede de computadores interligando as escolas municipais da cidade do Rio Grande, RS, e para o suporte a atividades de ensino nela desenvolvidas. É discutida uma solução baseada nos sistemas TCM e PostgreSQL para a modelagem e implementação do banco de dados demonstrando-se como podem ser adotadas boas práticas de engenharia de software aliadas a produtividade do processo de desenvolvimento através desta abordagem, através da escolha de um ferramental de apoio adequado.*

Abstract. *In this paper we present the use of free software on the development of a database to control a computers network for the public schools on the city of Rio Grande, RS, and to support the learning activities developed upon this network. We discuss a solution based on TCM and PostgreSQL for modeling and implementing the database system and we demonstrate how good practices on software engineering that also brings productivity can be adopted through this approach, by the selection of an adequate supporting toolset for the process.*

1. Introdução

Uma rede de computadores interligando um conjunto de escolas públicas é uma estrutura que implementa um ambiente de trabalho que possibilita a interação entre mais variados tipos de usuários. São alunos, professores, grupos de estudo, pais e outras pessoas da comunidade que são inseridos em um contexto no qual desenvolvem diversas atividades que dependem de um esquema robusto de controle sobre a própria rede e que produzem uma grande quantidade de informações que precisam ser armazenadas de forma clara e organizada. O presente artigo trata do desenvolvimento de um banco de dados para o controle e registro de informações na rede de computadores do **projeto ESCUNA** – Escola-Comunidade-Universidade – uma parceria entre a Fundação Universidade Federal do Rio Grande e a Prefeitura Municipal do Rio Grande, que busca a implementação da metodologia de projetos de aprendizagem e a informatização da rede municipal de ensino, visando a melhoria do ensino e a qualificação para o mercado de trabalho [Laurino e Rodrigues 2003].

Seguindo a filosofia do projeto ESCUNA, o banco de dados que dá suporte a rede de computadores foi inteiramente desenvolvido utilizando-se ferramentas de software livre, desde a modelagem até a implementação. Em particular, o banco de dados foi desenvolvido utilizando prioritariamente dois sistemas, o **TCM** [Wieringa 2003] para a modelagem e o banco de dados **PostgreSQL** [PostgreSQL 2003], além de ferramentas adicionais para automatização de tarefas

na fase de implementação. A seguir discutiremos os fatores que levaram a escolha destas ferramentas e os recursos oferecidos por elas que foram utilizados no contexto deste trabalho.

2. O projeto ESCUNA

Para que possamos compreender o contexto em que este banco de dados foi desenvolvido vejamos uma breve descrição de requisitos. O objetivo desta descrição é apresentar uma visão geral do cenário em que as soluções de software livre estão sendo aplicadas já que este artigo propõe-se a discutir os benefícios do uso das ferramentas de software livre na construção de sistemas de bancos de dados e não o desenvolvimento do projeto em si.

A rede do projeto ESCUNA foi estruturada para comportar um conjunto de 33 escolas da rede municipal da cidade do Rio Grande, RS, interligadas por uma rede metropolitana e dispondo cada uma de um laboratório de informática com uma rede local. Cada escola emprega uma equipe de trabalho, denominada "kit-escola",, que dá assistência aos usuários da rede: professores, alunos e demais pessoas da comunidade.

Além dos fatores organizacionais/tecnológicos, deve-se salientar que o projeto ESCUNA tem por objetivo a disseminação da metodologia de projetos de aprendizagem nas escolas da rede municipal da cidade do Rio Grande. Um projeto de aprendizagem é uma atividade de ensino auxiliada pelo uso de computadores na qual um grupo de participantes (professores e alunos) escolhe um tema para pesquisa e debates entre eles. Todo aluno ou grupo de alunos pode criar seus próprios projetos de aprendizagem e convidar outras pessoas para que participem deles. As informações produzidas durante o desenvolvimento dos projetos ficam disponíveis através da Internet podendo ser enriquecidas com as interações dos próprios participantes do projeto, bem como por contribuições de usuários externos. O sucesso deste sistema, e de todo o projeto, depende da capacidade do próprio sistema em prover meios eficientes de comunicação (trocas de experiências) entre alunos das diversas escolas envolvidas no projeto.

3. Modelagem do banco de dados

Para a modelagem do sistema foi utilizado o editor de diagramas TCM – *Toolkit for Conceptual Modeling* [Wieringa 2003]. Esta ferramenta é um software livre desenvolvido na Universidade de Tente, Holanda, que permite a edição de diagramas de modelagem simbólica oriundos de diversas ferramentas, com destaque para diagramas de análise estruturada e diagramas UML.

O TCM caracteriza-se por ser uma ferramenta simples e de fácil utilização. Ao contrário da maioria das ferramentas de modelagem, especialmente as de modelagem UML, o TCM apresenta uma interface de edição de propriedades e atributos totalmente baseada na própria área de edição do diagrama evitando, por exemplo, que o usuário precise trabalhar com uma grande quantidade de janelas para poder definir os parâmetros de seu modelo. Esta característica contribui para que o usuário consiga desenvolver rapidamente seus modelos. Contudo, a fim de evitar que o usuário estabeleça ligações impróprias em seus diagramas, o TCM também realiza um controle semântico sobre os diagramas em tempo de edição.

A modelagem do banco de dados teve como foco o desenvolvimento de modelos de estrutura estática (diagramas de classe da UML) para representar o modelo de dados. Foi também utilizado um conjunto de convenções definidas de acordo com a ferramenta de geração de código utilizada, que será discutida a seguir.

4. Implementação do banco de dados

Duas questões relativas à implementação do banco de dados merecem destaque: a escolha do sistema de bancos de dados utilizado e a ferramenta de geração automática de código.

4.1. O sistema de bancos de dados

O sistema de bancos de dados escolhido para a implementação do projeto foi o PostgreSQL [PostgreSQL 2003]. Esta escolha foi justificada pelas seguintes características oferecidas por este produto:

- **Conformidade com o modelo relacional.** Quase todas as características esperadas em um sistema de bancos de dados relacionais são encontradas no PostgreSQL, como consultas declarativas em SQL, controle de concorrência através de transações, gerenciamento de usuários e otimizações de consultas.
- **Mecanismos de controle de integridade.** Uma das limitações dos sistemas de bancos de dados livres tem sido a falta de mecanismos de definição de regras de integridade como integridade referencial (chaves estrangeiras), regras e gatilhos. As versões mais recentes do PostgreSQL tem oferecido mecanismos satisfatórios para atender à estas demandas.
- **Extensões do modelo objeto-relacional.** O PostgreSQL possui algumas características de orientação a objetos associada ao seu modelo relacional, em especial o recurso de herança.

4.2. Ferramenta de geração automática de código

Uma ferramenta bastante útil no desenvolvimento do banco de dados baseado em TCM e PostgreSQL foi o gerador de código **tcm2sql** [Zagrodnick 2003]. Esta ferramenta toma um diagrama de estrutura estática gerado pelo TCM e produz o código em Linguagem de Definição de Dados (DDL) SQL que implementa o modelo descrito no diagrama. Embora a modelagem seja baseada em diagramas de classe da UML, o tcm2sql propõe um conjunto de convenções sintáticas para os diagramas que lhe servem como entrada de dados que permitem a definição de diversas formas de restrições que podem ser implementadas utilizando-se os recursos em um sistema de bancos de dados. Na Figura 1 podemos ver dois exemplos destas regras.

O primeiro exemplo é a da definição de integridade referencial. No exemplo da Figura 1.a vemos duas tabelas, Aluno_Pos_Graduacao e Curso, que se relacionam através da chave cod_curso. Segundo a convenção do tcm2sql a representação da chave primária se dá prefixando-se o atributo com o símbolo *sustenido* (#), e a chave estrangeira é representada com o símbolo do *acento til* (~). Além destes a convenção ainda exige que seja representada a cardinalidade entre as classes e a indicação implícita da chave primária que servirá como chave estrangeira, na forma de uma indicação de *papel*.

O segundo exemplo mostra a definição de uma regra. De acordo com a convenção proposta pelo tcm2sql, a área na representação da classe destinada a declaração de parâmetros pode ser utilizada para a definição de restrições de valor para os atributos daquela classe. No exemplo da Figura 1.b, a classe Serie possui restrições quanto ao número da série (que não pode ser maior do que 8) e do nível (que não pode ser maior do que 3).

A ferramenta ainda oferece recursos para a definição de outros tipos de restrições como generalizações, agregações, relações do tipo *n-para-n*, ou mesmo definições de regras mais complexas.

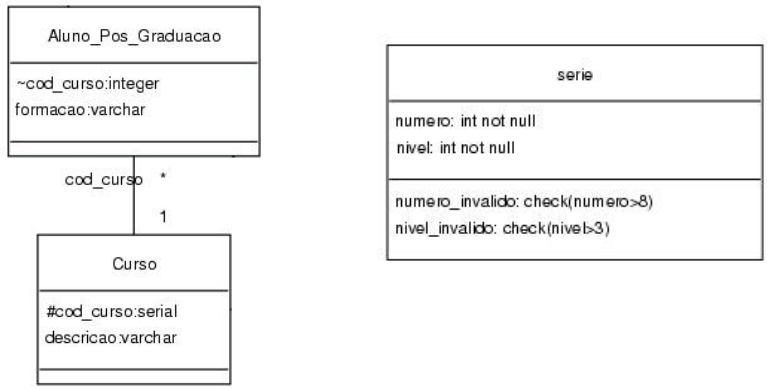


Figura 1. Exemplos de restrições definidos pelo tcm2sql.

5. Conclusões

Um dos principais fatores que inibem uma maior disseminação das soluções baseadas em sistemas de software livre em ambientes de produção, especialmente no desenvolvimento de sistemas baseados em bancos de dados, são a utilização boas práticas de engenharia de software aliada a produtividade.

Com o desenvolvimento deste banco de dados para a rede de computadores do projeto ESCUNA utilizando uma solução baseada em TCM/PostgreSQL, foi possível chegar-se a um modelo de trabalho que trouxe tanto a segurança de um processo de desenvolvimento coeso quanto a produtividade oriunda do uso de ferramentas automatizadas de apoio a implementação, tendo como resultado final um processo de desenvolvimento maduro e satisfatoriamente documentado.

Referências

Laurino, D., Rodrigues, S. C. (2003) "Projeto Escola-Comunidade-Universidade: buscando metodologias educativas, interativas e interconectivas em uma visão sistêmica", http://www.ead.furg.br/projetoescolas/, fevereiro.

PostgreSQL Global Development Group (2003) "PostgreSQL", http://postgresql.org/, maio.

Wieringa, R. (2003) "TCM – Toolkit for Conceptual Modeling", http://wwwhome.cs.utwente.nl/~tcm/, janeiro.

Zagrodnick, C. (2003) "tcm2sql", http://www.gocept.com/opensource/projects/tcm2sql, maio.

Distribuição GoboLinux

André Detsch, Guilherme B. Bedin

[1] GoboLinux.org

{detsch,gbedin}@gobolinux.org

Resumo: Este artigo descreve a evolução de uma proposta de uma hieraquia de diretórios inovadora, diferente da tradicionalmente utilizada em sistemas UNIX, para uma distribuição GNU/Linux completa, denominada GoboLinux. São apresentadas as características de uso da distribuição, bem como as facilidades proporcionadas pelo GoboLinux.

Abstract: This paper describes the evolution of a proposal for a innovative directory hierarchy, different than that traditionally used in UNIX systems, into a complete GNU/Linux distribution, called GoboLinux. The main peculiar characteristics of the distribution are presented, as well as the facilities provided by GoboLinux.

1 Introdução

O GoboLinux ([10]) é uma distribuição Linux cuja principal característica é o uso da hierarquia de diretórios de mesmo nome apresentada em [2]. A idéia inicial do projeto era apenas estabelecer uma nova hierarquia de diretórios que facilitasse o gerenciamento e o compartilhamento de programas compilados entre alguns usuários. Entretanto a aceitação dos conceitos apresentados foi tão boa que levou ao aperfeiçoamento das idéias e criação de facilidades de instalação, para que mais pessoas pudessem utilizá-las.

Ao longo de pouco mais de um ano, a distribuição GoboLinux evoluiu significativamente. Entretanto, tomou-se o cuidado quanto a divulgação da distribuição: esperou-se que a distribuição e a hierarquia atingissem um grau de maturidade que possibilitasse seu uso por um grande número de pessoas. Este anúncio em escala mundial foi cautelosamente postergado, mas o início de discussões mais intensas sobre uma estrutura de diretório para o Linux que fosse mais adaptada os uso em *desktop* ([6]), nos levou a trabalhar mais intensamente de forma a enfim divulgar em grande escala o projeto ([5, 7]). Após os anuncios, o GoboLinux deixou de ser um sistema usado somente pelos seus desenvolvedores. Com base no número de downloads da distribuição, relatos e sugestões dos novos usuários, que se inscreveram na lista do GoboLinux, o projeto continua crescendo e progredindo.

Este artigo, mais do que um relato de um ano de projeto GoboLinux, estabelece um marco entre uma idéia divulgada regionalmente e uma iniciativa capaz de influenciar mundialmente a utilização do sistema Linux. A Seção 2 descreve a hierarquia GoboLinux, lembrando os conceitos básicos de sua estrutura. Em seguida, são discutidos aspectos da distribuição GoboLinux (Seção 3). Por fim, a Seção 4 conclui o artigo, apresentado algumas metas e idéias ainda não implementadas.

2 Hierarquia GoboLinux

O objetivo da hierarquia GoboLinux é reestruturar a estrutura de diretórios Unix, para facilitar o gerenciamento de programas e entendimento do sistema, sem quebrar a compatibilidade com a árvore tradicional. Para isso, são utilizados três componentes básicos: o diretório `/Programs`,

o diretório `/System/Links` e um conjunto de links para compatibilidade com a árvore UNIX tradicional.

O diretório `/Programs` contém um subdiretório para cada programa instalado (*Gimp*, *FileUtils*, *XFree86*, *Qt*, etc.). Cada um destes subdiretórios possui, por sua vez, um subdiretório para cada versão do programa em questão assim como um link `Current` que aponta para o subdiretório contendo a versão em uso. Os programas ficam contidos, na sua totalidade, dentro do subdiretório da versão correspondente, onde tipicamente são criados subdiretórios `bin`, `lib`, `sbin`, `share`, `man` e `info` que contém os arquivos que, na estrutura tradicional de diretórios UNIX, seriam copiados para `/usr/bin` (ou `/bin`), `/usr/lib` (ou `/lib`) e assim por diante. Cada programa possui também um diretório Settings que armazena os arquivos de configuração referentes ao programa (arquivos que estariam tipicamente localizados no `/etc`). Note que este diretório é único para todas as versões do programa, de forma que sejam mantidas as configurações personalizadas do aplicativo em caso de upgrade ou downgrade.

O diretório `/System/Links` concentra links referentes à cada categoria de arquivo instalado dentro dos diretórios das versões dos programas. O `/System/Links/Executables`, por exemplo, contém links para todos os arquivos de todos os subdiretórios bin e sbin (executáveis) dentro de cada verão de cada programa instalado. Processo similar ocorre com as bibliotecas, headers manuais e arquivos compartilhados. Conforme será explicado na Seção 3, a gerência destes links é realizada de maneira totalmente automática.

A compatibilidade com o legado UNIX é obtida através da criação de links extras, como por exemplo, `/etc -> /System/Settings`, `/bin->System/Links/Executables` e `/lib -> /System/Links/Libraries`, que espelham a árvore GoboLinux na árvore UNIX. Estes links são necessários devido à existência de caminhos hardcoded em scripts, bibliotecas, executáveis e arquivos de configuração em geral ([3]). Entretanto, estes caminhos podem ser facilmente ocultados utilizando a ferramenta *GoboHide* ([9]).

3 Distribuição GoboLinux

O projeto GoboLinux teve seu início quando, baseando-se nos passos descritos pelo Linux From Scratch ([1]), compilou-se um sistema base utilizando uma nova estrutura de diretórios. Esta estrutura já vinha sendo discutida e parcialmente utilizada a alguns meses, durante os quais, sobre uma distribuição tradicional, novos programas iam sendo gradativamente compilados e estruturados de acordo com a nova estrutura de diretórios ([4]). Uma vez compilado um sistema totalmente estruturado dentro das nossas idéias, publicou-se um artigo sobre o assunto ([2]) e várias pessoas demonstraram interesse. Com o passar do tempo e a cada novo usuário, o GoboLinux passou a ter, gradativamente, toda a funcionalidade que faz com que, hoje, possa ser de fato considerada uma distribuição Linux completa. Esta seção apresenta algumas de suas principais características.

Gerenciamento de Pacotes

Conforme descrito na Seção anterior, cada versão de cada programa possui o seu próprio diretório. Isto possibilita que a gerência de pacotes binários se dê de forma extremamente simples e sem depender de banco de dados. O script "`CreatePackage`", responsável pela criação do pacote a partir de um programa instalado, basicamente compacta o diretório da versão indicada, criando um arquivo cujo nome contém o nome do programa, a versão e a arquitetura para qual foi compilado[1] (por exemplo, `Qt--3.1--i686.tar.bz2`). Porém, antes de realizar a compactação, o script identifica, com o auxílio do `ldd`, as dependências do pacote, gerando um arquivo "`.dependencies`" que lista de quais programas (e respectivas versões) o novo pacote depende.

[1] Atualmente, sempre i686.

A instalação de um pacote se dá através do script "`InstallPackage`", que descompacta o pacote dentro do diretório `/Programs` e chama, internamente, o script "`LinkProgram`", que cria os links referentes ao pacote dentro da estrutura `/System/Links`. Após, são identificadas as dependências pendentes, ou seja, quais programas do "`.dependencies`" não está instalado no sistema, informando o usuário sobre estas pendências. Caso exista um pacote referente a alguma dependência não satisfeita em algum diretório de uma lista pré-definida (contendo, entre outros, o diretório de pacotes do CD de instalação), o usuário é informado, e pode proceder com a instalação da dependência sem precisar de nenhum comando adicional.

A desintalação de pacotes é realizada pela simples remoção do diretório correspondente (por exemplo, `rm -rf /Programs/Qt/3.0.5`). Como resultado desta remoção, surgirão links quebrados dentro da estrutura `/System/Links`. Sua existência não compromete o funcionamento do sistema, mas são removidos automaticamente pelo script "`RemoveBrokenLinks`".

Compilação de Programas

A compilação de novos programas é automatizada através do script "`CompileProgram`". No caso de programas com arquivo de configuração de instalação (*configure*) gerado a partir do *GNU autoconf* (grande maioria dos softwares livres) o script utiliza o parâmetro `--prefix` para definir o destino dos arquivos sendo instalados. Por exemplo, no caso de se instalar o *Qt* 3.1, o script executará o comando `configure --prefix=/Programs/Qt/3.1/`. No caso de programas cuja instalação não permite a passagem deste parâmetro, o *configure* (ou o próprio *Makefile*) deve ser ajustado, automaticamente, através de scripts que buscam substituir os caminhos contidos no arquivo ou mesmo, em alguns casos específicos, manualmente. Uma vez criados (ou ajustados) os *Makefile*s, o CompileProgram prossegue a instalação rodando make (que compila os fontes) e make install (que copia/instala os arquivos compilados para o destino especificado). A seguir, é executado o script "`LinkProgram`", que cria os links na estrutura `/System/Links`.

Processo de Instalação

Inicialmente, a instalação do GoboLinux se dava através da copia do sistema base de uma máquina para outra. A partir do momento em que a base de usuários se tornou maior, foi criado um CD de instalação, que torna a instalação do GoboLinux uma tarefa bastante simples.

O CD de instalação é bootável, e o sistema executado é o próprio GoboLinux. O arquivo `/Depot/Docs/INSTALL` guia o processo de instalação, que é composto de algumas etapas bastante simples, citadas a seguir. Após iniciar o sistema, o usuário deve preparar uma partição para instalar o sistema e montá-la. A seguir, deve ser executado o script "`BaseInstall`" passando como parâmetro o ponto de montagem escolhido. Após esta execução, um sistema básico estará instalado, sendo necessário apenas ajustar os arquivos de configuração fstab, que descreve os drives e pontos de montagem utilizados pelo sistema, e o arquivo Options, arquivo de configuração dos bootscripts do GoboLinux (ajustando configurações de rede e teclado, por exemplo). Uma vez reinicializada a máquina, pode-se instalar pacotes adicionais disponíveis no próprio CD de instalação.

Sistema de Boot

O GoboLinux possui seu próprio sistema de configuração de *boot* (pacote `BootScripts`), onde os dois principais scripts fazem a maior parte de trabalho, `Init` e `Done`. Para lidar com os diferentes runlevels existentes, ainda existem os scripts `Multi` e `Single`. Estes arquivos são simplemente seqüências de comandos, prefixados pela palavra Exec e uma mesagem de texto. Por exemplo, uma parte do `Init`:

```
Exec "Setting clock..." SetClock
Exec "Loading keymap..." loadkeys "$KeymapLayout"
```

Tarefas mais sofisticadas, são definidas como funcões do shell no arquivo `Tasks`. Estas funções também podem ser chamadas pela linha-de-comando usando o script "`RunTask`". As configurações são definidas através de variáveis de ambiente no arquivo `Options`.

Documentação

Por se tratar de uma distribuição não-convencional, a documentação do GoboLinux é uma parte importante do projeto. A página do GoboLinux ([10]) possui, além de uma lista de discussão [2], FAQs e alguns HOW-TOs. Está em fase de desenvolvimento um manual completo sobre as funcionalidades do GoboLinux, com ênfase na utilização de seus scripts, que pode ser encontrado em [8].

4 Conclusões e Projetos Futuros

O GoboLinux é um projeto em franca expansão. Após a divulgação em sites mundialmente conhecidos, as discussões sobre o tema e a procura por informações tem aumentado bastante. Ao falar de projetos futuros, pode-se dividir o projeto em duas partes: O GoboLinux enquanto estrutura de diretórios alternativa e o GoboLinux enquanto distribuição.

A estrutura de ditetórios GoboLinux pode ser utilizada em conjunto com distribuições tradicionais, aproveitando o suporte e a disseminação das mesmas. Está em fase de implementação um *wrapper* de chamadas para o *kernel* que captura, durante a instalação de algum programa, tentativas de gravação em diretórios como `/usr/lib` e as adapta para o caminho adequado na estrutura GoboLinux. Este mecanismo, potencialmente, irá facilitar a adaptação de distribuições para a nova hierarquia.

Já a distribuição GoboLinux, além de servir de referência para a nova hierarquia, também introduz outras características peculiares, como o sistema de boot. Para o futuro, tais inovações irão continuar, de forma a adaptar, cada vez mais, o Linux à realidade de uso em computadores *desktop*.

Referências

[1] Gerard Beekmans. "*Linux From Scratch*", `http://www.linuxfromscratch.org`

[2] Hisham H. Muhammad, André Detsch, "*Uma nova proposta para a árvore de diretório Unix*", Workshop de Software Livre 2002, Porto Alegre, Maio 2002.

[3] Hisham H. Muhammad, Rafael Guterres Jeffman, "*Portabilidade e flexibilidade em software livre: a experiência do GoboLinux*", `http://gobolinux.org/documentation/wsl2003/`

[4] Hisham. H. Muhammad, *LodeLinux*, `http://www.cscience.org/~lode/computer/lodelinux.php`

[5] Hisham H. Muhammad, "*The Unix tree rethought: an introduction to GoboLinux*", `http://www.kuro5hin.org/story/2003/5/9/05015/62649`, `http://www.osnews.com/comment.php?news_id=3511`

[6] "*If I had my own Distro...*", `http://slashdot.org/article.pl?sid=03/04/29/1958212&mode=thread`

[7] "*GoboLinux Rethinks The Linux Filesystems*", `http://developers.slashdot.org/article.pl?sid=03/05/10/1636245`

[8] Leandro Motta Barros, "*The GoboLinux Magnificent Book*", `http://lmbarros.tripod.com/GoboLinux/GoboLMB-020030202.pdf`

[9] Lucas Correia Villa Real, Felipe W. Damasio, "*GoboHide*", `http://gobolinux.org/documentation/wsl2003/`

[10] Projeto GoboLinux, `http://www.gobolinux.org`

[2] `http://gobolinux.org/mailman/listinfo/gobo-l`

Portabilidade e flexibilidade em software livre: a experiência do GoboLinux

Hisham H. Muhammad, Rafael G. Jeffman

[1] GoboLinux.org

{lode,rafasgj}@gobolinux.org

Resumo. *Este artigo discute alguns dos problemas encontrados em diversas aplicações distribuídas como software livre, no que se refere a flexibilidade e portabilidade destas aplicações e as implicações das restrições encontradas na adaptação destas aplicações ao GoboLinux, uma distribuição Linux que não segue alguns padrões convencionais como a hierarquia de diretórios e o processo de inicialização. São abordadas algumas ferramentas utilizadas para o desenvolvimento de software livre e alguns problemas encontrados tanto nas ferramentas quanto no mau uso destas. São apontadas soluções possíveis e é proposta uma nova forma de encarar a flexibilidade e a portabilidade durante o desenvolvimento de software livre.*

Abstract. *This article discusses some of the problems existing in several applications distributed as free software, in terms of flexibity and portability and the implications of those restrictions in the adaptation of these applications to GoboLinux, a Linux distribution that does not follow some of the established conventions such as the directory hierarchy and the start-up process. Some of the tools used in the development of free software are covered in this work, as well as some of the problems that arise in their use and misuse. Possible solutions are pointed and, most importantly, a new way to view flexibility and portability in the development of free software.*

1 Introdução

O conceito de portabilidade, adquire, com o modelo de desenolvimento de software livre, novas proporções. Software proprietários normalmente são desenvolvidos visando as arquiteturas de hardware e software dominantes no mercado, e manter uma equipe de desenvolvimento e testes para arquiteturas menos utilizadas é inviável para a maioria das empresas, mesmo que na prática, garantir a portabilidade de um programa muitas vezes envolva apenas cuidados triviais. Com a disponibilidade do código-fonte, este tipo de verificação acaba sendo feita pela própria comunidade de usuários, trazendo benefícios para ambas as partes: por um lado, a comunidade de usuários potenciais da aplicação aumenta; por outro lado, a plataforma é beneficiada com a disponibilidade de mais aplicações.

Apesar disso, a maioria das questões referentes a portabilidade que são enfrentadas pelos usuários poderiam ser evitadas. Uma série de premissas assumidas pelos autores das aplicações, com base nas características dos sistemas utilizados por eles, não são verdadeiras em outros ambientes. Não estamos falando aqui de funcionalidades do sistema operacional que realmente demandariam bibliotecas de compatibilização, mas simplesmente de peculiaridades como, por exemplo, localização de arquivos.

Este artigo relata a experiência obtida nesse tema a partir do desenvolvimento do *GoboLinux*, uma distribuição GNU/Linux baseada em uma hierarquia de diretórios alternativa ([7]). A partir desta experiência, uma série de deficiências de portabilidade e flexibilidade freqüentemente ocorrentes em softwares livres foram encontradas. O artigo discute como estes problemas podem ser evitados, de modo a tornar a comunidade de desenvolvedores e usuários de software livre menos centrada em torno de um conjunto pequeno de distribuições, ou em última análise, de um único sistema operacional.

2 Padronização

O estabelecimento de padrões, à primeira vista, visa diminuir as diferenças entre os ambientes. De fato, tentativas de estabelecimento de padrões como o Linux Standard Base ([6]) têm este como seu objetivo, arbitrando características que os sistemas devem seguir de modo a buscar uma certa homogeneização. Padrões como estes acabam apenas por limitar as possibilidades, e restringir opções.

	Total de arquivos	Contendo *hard-coded*
`/System/Links/Executables`	1760	701 (39,82%)
`/System/Links/Libraries`	13878	3138 (22,61%)
`/System/Links/Headers`	6351	5 (0,07%)

Tabela 1: Quantidade de arquivos contendo caminhos hard-coded

Padrões devem, na verdade, ter por objetivo *especificar interoperabilidade*. Somente desta forma é possível manter a liberdade de implementação, para que alternativas melhores possam sempre ser experimentadas e, possivelmente, adotadas. Um caminho nesta direção é a adoção das GNU AutoTools ([8]) por um número crescente de aplicações. Ao utilizar estas ferramentas, o desenvolvedor permite que a aplicação seja utilizável em diversos ambientes, uma vez que o processo de detecção e configuração da aplicação para o ambiente-alvo é realizado pelo AutoConf.

O uso de GNU AutoTools por parte das aplicações permite que elas sejam utilizadas no GoboLinux sem adaptação alguma, o que facilitou largamente a instalação de aplicações. Ainda assim, é necessária uma ressalva em relação ao uso correto das ferramentas. Foi desenvolvido para o GoboLinux uma ferramenta chamada FIBOSANDBOX que realiza de forma transparente a instalação de aplicações utilizando um ambiente com permissões de escrita no sistema de arquivos restringidas. Através do FIBOSANDBOX, detectou-se que diversas aplicações, apesar de utilizarem GNU AutoTools para configurar o ambiente de compilação, não o fazem corretamente para a configuração dos caminhos de instalação.

3 Configurações

Em um grande número de programas, os autores cometem o erro de pressupor como fixas determinadas características que, especialmente dada a natureza do software livre, são flexíveis. O modelo de software livre estimula a modificação. Mesmo tendo desenvolvido um programa para um determinado sistema operacional, o autor deve estar ciente de que ele eventualmente será portado.

3.1 Tipo de ambiente

No passado, as grandes diferenças entre UNIX e VMS faziam com que um programa portável entre estas duas arquiteturas fosse facilmente portável para qualquer outra. Atualmente, com a popularização do GNU/Linux e da família BSD, a maior parte dos softwares livres visa apenas a plataforma UNIX. O surgimento do Cygwin ([1]), ambiente UNIX-like para Windows, tenta remediar esta situação. Infelizmente esta abordagem representa uma solução apenas para a plataforma Windows, e técnica é, de certa forma, invasiva, uma vez que de certa forma é o ambiente que é portado, e não a aplicação. O projeto Fink ([3]) lida com questões de portabilidade de softwares livres para o sistema operacional Mac OS X, que atualmente é oficialmente considerado um sistema UNIX. A necessidade da existência de projetos como o Fink apontam a pobre portabilidade existente em um grande número de aplicações, mesmo entre diferentes sistemas UNIX.

3.2 Caminhos no sistema de arquivos

O erro mais comum encontrado são os caminhos *hard-coded* no código-fonte dos programas. O GoboLinux possui uma constituição de caminhos totalmente diferente, o que demanda da aplicação um nível de portabilidade de caminhos similar à necessário para a *instalação* em ambientes não-UNIX. Devido à hierarquia de compatibilidade apresentada em [7], o GoboLinux assemelha-se, em execução, a um sistema UNIX tradicional. Através de opções como `prefix` e `sysconfdir`, as GNU AutoTools permitem o uso de caminhos arbitrários. A ferramenta FIBOSANDBOX, entretanto, detectou que uma série de aplicações não abstraem todos os caminhos referenciados pelas ferramentas.

Caminhos hard-coded dentro do código binário das aplicações, embora não caracterizem um problema de portabilidade sério para software livre (uma vez que o código-fonte é disponível), podem causar incoveniências de interoperabilidade entre sistemas arquiteturalmente compatíveis, como diferentes distribuições Linux. A tabela 1 demonstra a quantidade de caminhos hard-coded encontrados em uma instalação GoboLinux com 275 pacotes instalados, entre aplicações, ferramentas e bibliotecas. É importante

ressaltar que os números listados na tabela referem-se apenas caminhos hard-coded relativos ao caminho-alvo da compilação do pacote — isto é, apenas em relação a caminhos *hard-coded* baseados em `/Programs` — isoladamente de eventuais caminhos *default* incluídos nos código-fonte dos programas. Este tipo de avaliação não seria possível em uma distribuição Linux tradicional, uma vez que caminhos inseridos na compilação e caminhos *default* supridos pela aplicação se confundiriam. Os resultados mostram que caminhos hard-coded são um problema considerável para a flexibilidade de executáveis e bibliotecas, mas são um problema raro, mas ocorrente, em arquivos de *header*s C/C++.

3.3 Identificação do super-usuário

Tradicionalmente o usuário com privilégios em relação a acesso aos dispositivos e arquivos em sistemas multi-usuário (o super-usuário) tem sido chamado de `root` por convenção. Mas não existe uma regra que obrigue os sistemas a utilizar este nome, é apenas prática comum. O única restrição existente para o super-usuário é que este usuário possua um identificador 0 (zero). Para o sistema, o usário com identificador 0 (zero) é o super-usuário, não importando qual o nome seja dado para ele.

Muitos desenvolvedores assumem que o nome do super-usuário é `root`, o que não é uma boa prática, principalmente em relação à portabilidade. Outros sistemas compatíveis com UNIX, por exemplo o BeOS ([5]), não possuem um usuário `root`, mas possuem um super-usuário. Programas escritos de forma a não assumir um usuário `root`, mas identificando o usuário pelo seu identificador numérico, tem menos restrições para que possam ser portados para outras plataformas.

A utilização de outros nomes ao invés de `root` apresenta algumas vantagens como nomes mais representativos, menos ambíguos (usuário `root`, diretório `/root`, sistema de arquivos ROOTFS), e mais seguros, uma vez que é pratica comum tentar invadir sistemas utilizando o login do usuário `root`.

Pelos motivos citados existem diversas vantagens na utilização de nomes alternativos para o super-usuário, tanto para a administração do sistema, quanto para o desenvolvimento de aplicações, neste caso não assumindo um nome pré-definido. No GoboLinux, o nome do super-usuário é `gobo`, e este pode ser facilmente alterado pelo usuário.

3.4 Scripts de inicialização

Não existe um padrão para scripts de inicialização. As diferentes distribuições Linux utilizam uma série de variantes dos dois principais modelos históricos de *boot scripts* (System V e BSD). Deste modo, uma aplicação não deve assumir nada a respeito destes scripts. Ainda assim, uma série de pacotes distribuem scripts para inclusão de serviços na seqüência de inicialização do sistema. Distribuir estes scripts como uma conveniência não é um problema. Todavia, realizar a instalação destes scripts durante o processo padrão de instalação do programa é extremamente problemático, além de prejudicar a manutenção do próprio programa, uma vez que ele estará sujeito às mudanças ocorridas, por exemplo, até em diferentes versões de uma mesma distribuição. Além disso, existem diversos projetos para bootscripts alternativos, não associados a nenhuma distribuição específica, por exemplo baseados em dependências entre os serviços a serem inicializados ([4]).

Outro aspecto a ser levado em conta nos scripts de inicialização é o tipo de aplicação ao qual a máquina seráestinada. Tipicamente, servidores precisam de um número de serviços bem maior do que em máquinas utilizadas como estações de trabalho. Em um servidor o tempo gasto com a inicializaçõa do sistema não é tão crítico, uma vez que são máquinas das quais se espera um uso ininterrupto (24x7). O mesmo não acontece em estações de trabalho, onde se espera um tempo de inicialização bem menor. Para este tipo de máquina o número de serviços executados é bem menor e não faz sentido a utilização de grandes scripts como os BSD ou a complexidade do System V.

4 Ferramentas para compilação de programas

4.1 Libtool

O trabalho de desenvolvimento do GoboLinux detectou uma falha fundamental de projeto da ferramenta LIBTOOL, parte das GNU AutoTools. A falha se origina do fato da ferramenta ter sido desenvolvida tendo em mente os sistemas UNIX, que possuem uma série de repositórios centralizados de bibliotecas (`/lib`,

`/usr/lib`, `/usr/local/lib`), contendo uma série de bibliotecas de diversas procedências. O programa `libtool`, ao gerar um arquivo de dependência para bibliotecas durante a sua compilação (arquivo de extensão `.la`), especifica, neste arquivo, os caminhos das bibliotecas utilizadas na *compilação* da biblioteca nova. Isto é uma prática errada, pois referencia, no sistema onde a biblioteca será instalada, caminhos que possivelmente só existem no sistema onde a biblioteca foi compilada. Em tempo de execução, estes arquivos não são relevantes, pois o carregador de bibliotecas dinâmicas `ld.so` procura as bibliotecas com base em uma variável de ambiente (`LD_LIBRARY_PATH`), a solução mais portável.

4.2 Imake

Uma das ferramentas de compilação de programas com uma das maiores bases de instalação existentes é o IMAKE ([2]), uma vez que faz parte da distribuição do XFREE86, a implementação livre do X WINDOW SYSTEM, o ambiente gráfico padrão do mundo UNIX. Esta ferramenta, apesar de haver sido criada com a finalidade de melhorar a portabilidade do X WINDOW SYSTEM, possui um dos mais sérios problemas de flexibilidade existentes: ela assume que a aplicação sendo compilada será instalada dentro da mesma hierarquia de diretórios que o XFREE86. Isto limita em muito a sua aplicabilidade, e por esta razão o IMAKE raramente é utilizado em novos projetos. Entretanto, existe um série de programas importantes no mundo do software livre que o utilizam. Em uma lista de discussão dedicada à questões de projeto do XFree86[1], este foi também apontado recentemente como um problema.

5 Conclusão

Este artigo descreveu os principais problemas relativos à portabilidade e flexibilidade em software livre enfrentados durante o desenvolvimento da distribuição GNU/Linux GoboLinux. Além de apontar soluções, espera-se que este artigo possa ser utilizado como referência pelos desenvolvedores de software livre de modo que estas limitações sejam reduzidas e que a interoperabilidade do software livre em geral seja aumentada. É importante notar que todas estas questões surgiram dentro de uma distribuição Linux, utilizando uma arquitetura x86, o que leva a crer que problemas de portabilidade em diferentes arquiteturas de hardware, utilizando bases de software alternativas seja ainda maior. Características peculiares da distribuição GoboLinux tornaram possível realizar uma avaliação da disseminação de caminhos *hard-coded* no sistema.

O artigo apresentou ainda uma ferramenta livre incluída no GoboLinux denominada FIBOSANDBOX que permite o isolamento das permissões de escrita a uma área restrita sem o isolamento das permissões de leitura como ocorre normalmente com `chroot`. Ferramentas como esta apontam um caminho onde a flexibilidade do software em relação a diferentes ambientes possa testada de forma automática. A instalação de aplicações no ambiente GoboLinux pode ser considerada também uma forma de emulação da instalação em ambientes não-UNIX, o que caracteriza uma forma nova de teste de portabilidade.

Referências

[1] Cygwin Information and Installation. `http://www.cygwin.com`

[2] Dubois, Paul. *Software Portability with Imake*, O'Reilly & Associates, 1996.

[3] Projeto Fink. `http://fink.sourceforge.net`

[4] Gooch, R. Linux Boot Scripts. `http://www.atnf.csiro.au/people/rgooch/linux/boot-scripts/`

[5] Hacker, S. et al. *The BeOS Bible*. Peachpit Press, 1999.

[6] Linux Standards Base. `http://www.linuxbase.org`

[7] Muhammad, H. Detsch, A. *"Uma nova proposta para a árvore de diretórios UNIX"*, III Workshop de Software Livre, Porto Alegre, 2002.

[8] Vaughan, G., Elliston, B., Tromey, T., Taylor, I. *"GNU Autoconf, Automake and Libtool"*, New Riders Publishing, 2000.

[1] `forum@xfree86.org`